

EGO_X: EGOCENTRIC CROSS-EMBODIMENT MANIPULATION WITH EMBODIMENT DREAMING

Anonymous Authors

<https://egox-embodiment.github.io>

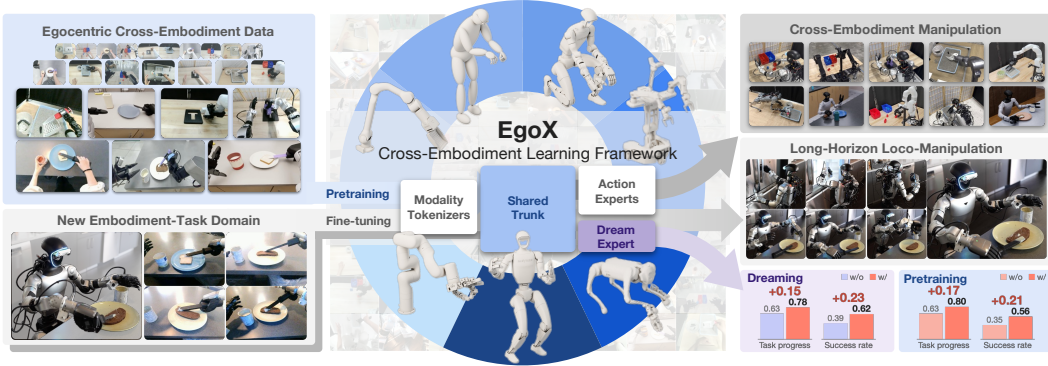


Figure 1: **Overview of EgoX.** EgoX learns from highly diverse embodiments, including humans, a humanoid, a wheeled bimanual robot, a quadruped, tabletop arms, and a simulated humanoid, with dexterous hands, grippers, and tools as end-effectors. It first pretrains on egocentric cross-embodiment data from humans and robots (top left), then fine-tunes on a new embodiment-task domain (bottom left). Its model, MXT+, routes each embodiment’s observations through modality tokenizers into a shared Transformer trunk and decodes actions with embodiment-specific action experts, while a training-only dreaming expert predicts future embodiment latents (center). The resulting policies perform manipulation across embodiments and long-horizon loco-manipulation on a humanoid (right, top and middle). Embodiment dreaming and cross-embodiment pretraining both improve the average task progress score and success rate (right, bottom).

ABSTRACT

Scaling robot manipulation requires learning from demonstrations collected across humans and heterogeneous robots. These sources cover diverse objects, scenes, tasks, and interaction strategies, but gaps in sensing, kinematics, dynamics, and action spaces make it difficult to train a shared policy effectively. We introduce EgoX, an egocentric cross-embodiment framework for robot manipulation. Its central design is to share manipulation structure at the level of semantically corresponding modalities, while preserving embodiment-specific sensing and control interfaces. EgoX stores human, real-robot, and simulated demonstrations as unified egocentric observation-action trajectories, pretrains a Modularized Cross-Embodiment Transformer with Embodiment Dreaming (MXT+) on the mixture, and fine-tunes it on target-embodiment demonstrations. MXT+ pairs embodiment-specific modality tokenizers and action experts with a shared Transformer trunk, and its training-only embodiment dreaming objective predicts future embodiment latents from egocentric visual and proprioceptive features, so that the trunk captures each embodiment’s dynamics. Across 13 task-embodiment pairs spanning five tasks and six robot embodiments, including a simulated humanoid, cross-embodiment pretraining raises average normalized task progress from 0.63 to 0.80 over training from scratch (0.53 to 0.77 out of distribution). Broader pretraining mixtures strengthen transfer, and embodiment dreaming further raises task progress from 0.63 to 0.78 across seven settings (0.56 to 0.74 out of distribution) and reduces mean cross-embodiment latent-geometry distance by 38.52%.

1 INTRODUCTION

Large-scale pretraining has driven rapid progress in natural language processing (Devlin et al., 2019; Brown et al., 2020) and computer vision (Radford et al., 2021; Dosovitskiy et al., 2021). Extending this paradigm to robotic manipulation requires accounting for the diversity of physical capabilities needed across tasks and environments. Tabletop arms, wheeled mobile manipulators, humanoids, and quadrupeds offer different combinations of mobility, reach, and workspace access, while grippers, dexterous hands, and specialized tools support different forms of object interaction. We envision a future populated by diverse robot embodiments, with broad capabilities and complementary strengths across tasks and environments (Pfeifer et al., 2007; Baines et al., 2024; Kirschner et al., 2025). However, collecting demonstrations and learning manipulation skills separately for every embodiment and task is costly and limits the reuse of experience across platforms. How can we collect and jointly use experience across these embodiments to develop shared manipulation knowledge while allowing each to adapt to its own morphology and domain?

Several complementary data sources offer a path toward this goal. Cross-embodiment robot learning reuses demonstrations collected across different bodies, sensors, and action spaces (O’Neill et al., 2024; Octo Model Team et al., 2024; Kim et al., 2025; Doshi et al., 2025; Wang et al., 2024; Gemini Robotics Team et al., 2025a). Egocentric human data captures diverse physical experience in real-world settings beyond robot deployment environments (Kareer et al., 2025; Niu et al., 2025; Qiu et al., 2025; Zheng et al., 2026). Simulation provides large-scale data for policy pretraining and, in some settings, zero-shot transfer (Deshpande et al., 2026; NVIDIA et al., 2025). Simulated demonstrations, including those from digital cousins of real tasks, can also be combined with real-world data for co-training (Maddukuri et al., 2025; Lin et al., 2026b). Yet combining these sources for low-level manipulation requires reconciling differences in morphology, end-effectors, proprioception, viewpoints, action semantics, and motion dynamics. An open challenge is to make these heterogeneous demonstrations jointly useful for manipulation across robots with substantially different bodies and end-effectors, including those performing legged loco-manipulation.

Recent work has examined scaling laws and data composition for robot manipulation (Lin et al., 2025; 2026a; TRI LBM Team et al., 2025). Alongside these questions of scale and data composition, we investigate how representations and learning interfaces can make heterogeneous experience useful for shared training. We treat humans, real robots, and simulated agents as embodiments on equal footing, each with distinct observations, actions, and temporal constraints. We aim to learn shared manipulation structure while preserving each body’s sensing and control interfaces.

We introduce EgoX (Fig. 1), an egocentric cross-embodiment framework built around a single methodological premise: transfer should occur through semantically corresponding modalities, while sensing and control interfaces remain embodiment-specific. The framework has three components. First, EgoX organizes human, real-robot, and simulated demonstrations in a unified observation-action representation, expressing poses in an embodiment-centric reference frame and grouping heterogeneous signals into anatomically or functionally relevant modalities. Second, MXT+, a Modularized Cross-Embodiment Transformer, places embodiment-specific modality tokenizers and action experts around a shared Transformer trunk, allowing heterogeneous embodiments to update a common representation without forcing their raw observations or action spaces to match. Third, MXT+ uses *embodiment dreaming*, an auxiliary objective that predicts future embodiment latents from the current shared representation. This training-only branch encourages the trunk to encode how each embodiment’s visual and proprioceptive state evolves, without requiring pixel reconstruction or deployment-time computation.

We evaluate EgoX on five manipulation tasks, from long-horizon loco-manipulation to precise insertion and tool use (Kitchen-Bread, Kitchen-Cleanup, Sort, Insert, and Scoop), across six robot embodiments: G1-Dex5, xArm7-Dex4, G1-Sim-Dex3, LocoMan-Dex2, R1Lite-Gripper, Z1-Tool. Human demonstrations provide an additional pretraining source. Our experiments compare ACT (Zhao et al., 2023), $\pi_{0.5}$ (Black et al., 2025a), and MXT+ variants across all tasks, and ablate the pretraining mixture, embodiment dreaming objective, and MXT+’s modular design. Cross-embodiment pretraining raises MXT+’s average task progress from 0.63 to 0.80, the best of all methods ($\pi_{0.5}$: 0.50, ACT: 0.37), and makes it the top method on 12 of 13 task-embodiment pairs. Broader pretraining mixtures transfer better, and embodiment dreaming raises

task progress from 0.63 to 0.78 across seven settings while making the trunk’s latent geometry more similar across embodiments. In summary, our contributions are:

1. **A unified, modality-structured egocentric representation for cross-embodiment robot manipulation.** Our framework, EgoX, places human, real-robot, and simulated demonstrations in a common observation-action format while retaining embodiment-specific sensing, action semantics, and validity masks.
2. **MXT+, a modular cross-embodiment Transformer with embodiment dreaming.** MXT+ places embodiment-specific tokenizers and action experts around a shared trunk and predicts future embodiment latents as an auxiliary training objective, to model each embodiment’s transition dynamics, with no dreaming computation at deployment time.
3. **A systematic study of cross-embodiment transfer.** Experiments across five tasks and six robot embodiments show that cross-embodiment pretraining benefits downstream robot manipulation, that broader embodiment mixtures improve transfer, and that embodiment dreaming improves the learned embodiment latent representations and provides further gains on task performance.

2 RELATED WORK

Cross-Embodiment Learning for Robotic Manipulation. Cross-embodiment robot learning aims to train policies that benefit from data collected on different bodies, sensors, and control interfaces. Large robot datasets and generalist policies demonstrate the value of scaling policy learning across tasks and robot embodiments (O’Neill et al., 2024; Kim et al., 2025; Black et al., 2025b;a; Gemini Robotics Team et al., 2025b), and recent work extends such pretraining to include human data (Luo et al., 2026a; NVIDIA et al., 2025). Prior methods handle embodiment heterogeneity with shared-trunk policies that use embodiment-specific tokenizers or action heads (Doshi et al., 2025; Niu et al., 2025; Wang et al., 2024), with a single embodiment-conditioned action head (Li et al., 2026b), or with image editing that aligns robot appearance across embodiments (Chen et al., 2024; Lepert et al., 2025). Existing methods either cover a relatively narrow range of embodiments and end-effectors or preserve heterogeneity by giving each embodiment a separate interface, leaving the extent of modality-level sharing unclear. In contrast, EgoX covers highly heterogeneous embodiments and end-effectors: it is evaluated on humanoid, wheeled bimanual, tabletop-arm, quadrupedal, and simulated platforms, with three- to five-fingered dexterous hands, two-finger and parallel-jaw grippers, and a tool. MXT+ combines embodiment-specific interfaces with modality-level sharing through modular tokenizers and action experts, and adds embodiment dreaming as a training signal through the prediction of future embodiment latents.

Robot Learning from Egocentric Data. Egocentric human data offers a scalable source of manipulation experience spanning diverse tasks, objects, and environments. Recent advances in wearable devices, such as XR headsets and smart glasses, have improved the quality and scale of such data (Chen et al., 2025; Hoque et al., 2026; Punamiya et al., 2026). Human videos have been used to learn transferable priors through visual, affordance, and latent-action objectives (Nair et al., 2023; Srirama et al., 2024; Ye et al., 2025; Bu et al., 2025; Zhang et al., 2026). Other methods use human demonstrations directly for policy learning, through domain adaptation (Punamiya et al., 2025), pre-training or co-training (Kareer et al., 2025; Yuan et al., 2025; Yang et al., 2025; Zheng et al., 2026; Shi et al., 2026; Zhu et al., 2026; Bi et al., 2026; Wei et al., 2026a), and transfer without additional target-robot demonstrations (Xu et al., 2026). Much of this literature emphasizes transferring human experience to target robot embodiments, while EgoX focuses on collecting and jointly learning from egocentric demonstrations across human and heterogeneous robot embodiments, and investigates how this shared training benefits manipulation across diverse platforms.

World Action Models and Embodiment Modeling. World models learn to predict how the world evolves under actions (Assran et al., 2025; Goswami et al., 2025; Gao et al., 2026). World action models couple such prediction with policy learning by jointly predicting actions and future observations, with targets including pixels, pretrained visual features, and 3D motion flow (Li et al., 2026a), embeddings from frozen or separately pretrained encoders (Zheng et al., 2025; Luo et al., 2026b), and future tactile signals or tactile latents (Niu et al., 2026; Lou et al., 2026). These works show that predicting future observations provides a useful training signal for policy learning. Other methods model the embodiment itself, either by predicting its dynamics or kinematics (Fu et al., 2025;

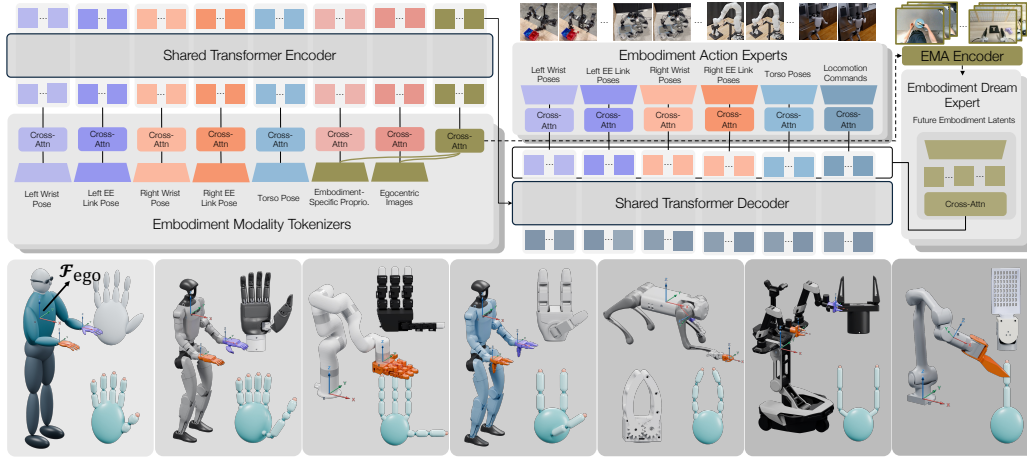


Figure 2: **MXT+ architecture and EgoX embodiments.** *Top:* embodiment-specific modality tokenizers map each embodiment’s observations (egocentric images, embodiment proprioception, and wrist, end-effector-link, and torso poses) into tokens that a shared Transformer encoder-decoder trunk fuses across embodiments. Embodiment action experts decode the trunk output into embodiment-specific action chunks, while embodiment-dreaming experts predict future embodiment latents supervised by EMA copies of the corresponding modality tokenizers. *Bottom:* the embodiments used in EgoX, from left to right: Human-Dex5, G1-Dex5, xArm7-Dex4, G1-Sim-Dex3, LocoMan-Dex2, R1Lite-Gripper, and Z1-Tool. Each panel marks the unified egocentric frame $\mathcal{F}_{\text{ego}}$ in which wrist and end-effector poses and actions are expressed, together with its wrist frames. Left and right end-effectors are shown in purple and orange, matching the modality colors above, alongside a close-up of each end-effector and its end-effector-link keypoints.

Patel & Song, 2025), or by encoding its structure through body graphs, canonical morphology representations, or embodiment-specific functional components in action generation (Sferrazza et al., 2025; Wei et al., 2026b; Li et al., 2026b). EgoX combines the ingredients of world action models and embodiment modeling through embodiment dreaming: alongside actions, MXT+ predicts future embodiment latents from egocentric visual features, robot joint configurations, and end-effector joint configurations using exponential-moving-average copies of its own modality encoders. The objective helps the shared trunk learn embodiment-aware dynamics while leaving the deployment-time action policy unchanged.

3 EGOX: EGOCENTRIC CROSS-EMBODIMENT MANIPULATION

EgoX makes three key design decisions to enable effective learning across diverse embodiments. First, we represent the heterogeneous data in a unified structure based on shared modalities within the unified egocentric frame (Sec. 3.1). To handle diverse sensing and control interfaces with different robots while facilitating positive transfer, EgoX leverages MXT+ (Fig. 2), where embodiment-specific tokenizers map observations into tokens and the transformer backbone can learn from a semantically aligned representation of diverse embodiments (Sec. 3.2). Finally, the auxiliary objective of *embodiment dreaming* encourages the transformer to learn each embodiment’s dynamics (Sec. 3.3). EgoX uses a two-stage training pipeline: pretraining on mixed cross-embodiment data, and fine-tuning on the target embodiment task (Sec. 3.4).

3.1 EGOCENTRIC CROSS-EMBODIMENT DATA REPRESENTATION

Although embodiments differ substantially as whole bodies, many of their parts play corresponding roles. We exploit this by decomposing each embodiment into anatomically or functionally corresponding modalities and aligning them in a unified egocentric frame. Formally, EgoX collects demonstrations from a set of embodiments $\mathcal{E}$ that includes both humans and robots. For each embodiment $e \in \mathcal{E}$, the dataset

$$\mathcal{D}_e = \{\tau_i^e\}_{i=1}^{N_e}, \quad \tau_i^e = (\mathbf{o}_{i,t}^e, \mathbf{a}_{i,t}^e)_{t=1}^{T_i^e}, \quad (1)$$

contains N_e trajectories. Trajectory τ_i^e has length T_i^e . The terms $\mathbf{o}_{i,t}^e$ and $\mathbf{a}_{i,t}^e$ are the observations and actions at time step t . Both are tuples of named modalities, defined below. When the trajectory is clear from context, we omit the index i .

Unified egocentric frame. To retain a consistent egocentric spatial interpretation, EgoX follows a unified egocentric frame $\mathcal{F}_{\text{ego}}$ (as shown in Fig. 2) attached to the egocentric main camera frame or the rigid body carrying the main camera, and expresses wrist and end-effector poses relative to it. At reset, the x -axis points forward toward the workspace, the y -axis points left, and the z -axis points upward. Poses streamed from teleoperation, sensing, or control interfaces are converted into this frame before data saving and model inference.

Observations and action targets. EgoX represents data from different embodiments in a unified schema of images, and proprioceptive states and actions of shared modalities. As depicted in Fig. 2, each demonstration is represented with the same observation-action dictionary structure:

$$\mathbf{o}_t^e := \{\mathbf{I}_{\text{main},t}^e, \mathbf{x}_{\text{torso},t}^e, \mathbf{x}_{\text{L/R wrist},t}^e, \mathbf{p}_{\text{L/R ee link},t}^e, \dots\}, \quad (2)$$

$$\mathbf{a}_t^e := \{\mathbf{x}_{\text{torso},t}^{e,\text{target}}, \mathbf{x}_{\text{L/R wrist},t}^{e,\text{target}}, \mathbf{p}_{\text{L/R ee link},t}^{e,\text{target}}, \dots\}. \quad (3)$$

Across embodiments, observations include egocentric main-camera images $\mathbf{I}_{\text{main},t}^e$, torso pose proprioception $\mathbf{x}_{\text{torso},t}^e$, wrist pose proprioception $\mathbf{x}_{\text{L/R wrist},t}^e$, and end-effector link pose proprioception $\mathbf{p}_{\text{L/R ee link},t}^e$. There may also be additional embodiment-specific streams such as wrist-camera images, robot joint states, and force feedback. Action targets are represented in the same modular format, including torso pose commands $\mathbf{x}_{\text{torso},t}^{e,\text{target}}$, end-effector pose commands $\mathbf{x}_{\text{L/R wrist},t}^{e,\text{target}}$, end-effector link pose commands $\mathbf{p}_{\text{L/R ee link},t}^{e,\text{target}}$, and when available, velocity commands.

The end-effector link poses $\mathbf{p}_{\text{L/R ee link},t}^e$ are task-relevant 3D keypoints on each end-effector, expressed in its local frame, such as the fingertips of human and robot hands and the contact points of grippers and tools (bottom of Fig. 2). Because they describe end-effector geometry rather than actuator joint coordinates, hands, grippers, and tools with different kinematics are expressed in comparable geometric terms. In this unified format, human and robot demonstrations alike become egocentric trajectories that pair visual observations with torso, wrist, and end-effector-link motion.

Shared modality vocabulary. Let $\mathcal{M} = \mathcal{M}_{\text{obs}} \cup \mathcal{M}_{\text{act}}$ be the modality vocabulary shared by all embodiments, indexed by m : each entry of $\mathbf{o}_t^e$ and $\mathbf{a}_t^e$ above, with L and R counted separately, is one modality, written $\mathbf{o}_t^{e,m}$ or $\mathbf{a}_t^{e,m}$. Not every embodiment provides every modality, so $\mathcal{M}_{\text{obs}}^e \subseteq \mathcal{M}_{\text{obs}}$ and $\mathcal{M}_{\text{act}}^e \subseteq \mathcal{M}_{\text{act}}$ denote the modalities available for embodiment e . Missing modalities are handled with mask tokens and loss masking, so token sequences remain semantically aligned across embodiments (Appendix A.1).

3.2 MODULARIZED CROSS-EMBODIMENT TRANSFORMER

To learn from heterogeneous human and robot demonstrations, EgoX uses MXT+, a Modularized Cross-Embodiment Transformer with Embodiment Dreaming. Building on the modular tokenization and action decoding of MXT (Niu et al., 2025), MXT+ separates embodiment-specific sensing and control interfaces from a shared policy backbone: observations are mapped to modality tokens, fused by a shared encoder-decoder Transformer trunk, and decoded into action chunks and future embodiment latents (Sec. 3.3) by modular experts.

Architecture overview. For each embodiment $e \in \mathcal{E}$, the MXT+ policy is the composition

$$\pi^e = \Psi^e \circ G_\theta \circ \Phi^e, \quad \Phi^e = \{\phi_m^e\}_{m \in \mathcal{M}_{\text{obs}}^e}, \quad \Psi^e = \{\psi_m^e\}_{m \in \mathcal{M}_{\text{act}}^e}, \quad (4)$$

where ϕ_m^e is the tokenizer of observation modality m , ψ_m^e is the action expert of action modality m , and G_θ is the shared trunk. The tokenizers Φ^e and experts Ψ^e form the paired input and output interfaces of embodiment e and are instantiated only for the modalities that e provides. Because all modalities come from the shared vocabulary $\mathcal{M}$ and missing ones are filled with mask tokens, G_θ receives the same modality layout from every embodiment. Cross-embodiment transfer therefore happens in the shared trunk, while Φ^e and Ψ^e absorb differences in sensing and control. Given an observation $\mathbf{o}_t^e = (\mathbf{o}_t^{e,m})_{m \in \mathcal{M}_{\text{obs}}^e}$, the policy predicts an action chunk, i.e., the next H action targets of every available action modality:

$$\hat{\mathbf{A}}_t^e = \pi^e(\mathbf{o}_t^e) = (\hat{\mathbf{A}}_t^{e,m})_{m \in \mathcal{M}_{\text{act}}^e}, \quad \hat{\mathbf{A}}_t^{e,m} = (\hat{\mathbf{a}}_{t+k}^{e,m})_{k=0}^{H-1} \in \mathbb{R}^{H \times d_m}, \quad (5)$$

where d_m is the dimension of action modality m . Appendix A.2 provides details on each module.

3.3 EMBODIMENT DREAMING

Behavioral cloning supervises the policy only through demonstrated actions; it does not require the policy to anticipate how the embodiment’s own state evolves. MXT+ therefore adds *embodiment dreaming*, an auxiliary objective in which the policy predicts future *embodiment latents*, i.e., compact representations of selected future observation modalities of the active embodiment. This encourages the shared trunk to capture embodiment-specific dynamics. At the same time, supervising in latent space lets the trunk predict future embodied state at the feature level, including visual and joint configurations, without reconstructing high-dimensional observations such as images.

Embodiment latent. For embodiment e , let $\mathcal{M}_{\text{dream}}^e \subseteq \mathcal{M}_{\text{obs}}^e$ be the observation modalities used as dreaming sources: the egocentric image, robot joint configuration, and end-effector joint configuration, whichever of them e provides. Robot joint configurations are not part of the action space, but they carry embodiment-specific state that is informative about body dynamics. An embodiment tokenizer ϕ_{dream}^e adds positional embeddings to the encoded features of each modality in $\mathcal{M}_{\text{dream}}^e$ and aggregates them by cross-attention from L_{dream} learnable queries into L_{dream} latent tokens of dimension d ($L_{\text{dream}} = 8$ in our experiments). These tokens are retained without pooling. The dreaming target at a future offset k is computed by exponential-moving-average (EMA) teacher copies of the policy’s modality tokenizers ϕ_m^e (Sec. 3.2) and of ϕ_{dream}^e , marked by a bar:

$$\mathbf{Y}_{t+k}^e = \text{sg} \left[\bar{\phi}_{\text{dream}}^e \left(\left[\bar{\phi}_m^e(\mathbf{o}_{t+k}^{e,m}) \right]_{m \in \mathcal{M}_{\text{dream}}^e} \right) \right], \quad (6)$$

Here $\mathbf{Y}_{t+k}^e = [\mathbf{y}_{t+k,j}^e]_{j=1}^{L_{\text{dream}}} \in \mathbb{R}^{L_{\text{dream}} \times d}$ is the sequence of target tokens, and $\text{sg}[\cdot]$ is the stop-gradient operator. As in the JEPA model family (Assran et al., 2025), the stop-gradient prevents the dreaming loss from updating the target, and the EMA update in Eq. (9) makes the target evolve slowly.

Dreaming expert. A dreaming expert ψ_{dream}^e reads the same trunk output as the action experts and predicts the corresponding latent-token sequences at a set of future offsets $\mathcal{K}$:

$$(\hat{\mathbf{Y}}_{t+k}^e)_{k \in \mathcal{K}} = (\psi_{\text{dream}}^e \circ G_\theta \circ \Phi^e)(\mathbf{o}_t^e). \quad (7)$$

Dreaming targets are sampled from future observations by the same data pipeline as action chunks.

Latent prediction loss. Each predicted token is supervised against its corresponding target token with a cosine alignment loss and an optional magnitude loss, averaged over future offsets and token indices:

$$\begin{aligned} \mathcal{L}_{\text{dream}}^e = & \frac{1}{|\mathcal{K}|L_{\text{dream}}} \sum_{k \in \mathcal{K}} \sum_{j=1}^{L_{\text{dream}}} \left[1 - \cos(\hat{\mathbf{y}}_{t+k,j}^e, \mathbf{y}_{t+k,j}^e) \right. \\ & \left. + \beta \ell_{\text{smooth-L1}}(\|\hat{\mathbf{y}}_{t+k,j}^e\|_2, \|\mathbf{y}_{t+k,j}^e\|_2) \right], \end{aligned} \quad (8)$$

where $\beta \geq 0$ weights the magnitude term.

EMA target update. Each teacher $\bar{\phi}$ in Eq. (6) tracks its student ϕ , i.e., ϕ_{dream}^e and ϕ_m^e for $m \in \mathcal{M}_{\text{dream}}^e$. After each optimizer step,

$$\bar{\phi} \leftarrow \alpha \bar{\phi} + (1 - \alpha) \phi, \quad \alpha \in (0, 1), \quad (9)$$

where ϕ and $\bar{\phi}$ denote module parameters.

3.4 CROSS-EMBODIMENT PRETRAINING AND TARGET FINE-TUNING

EgoX first pretrains MXT+ on a mixture of human and robot embodiments with action prediction and embodiment dreaming, and then fine-tunes it on demonstrations of the target embodiment.

Training objective. For a time step t of a trajectory of embodiment e , the action loss is the chunked behavioral cloning loss over the available action modalities,

$$\mathcal{L}_{\text{act}}^e = \sum_{m \in \mathcal{M}_{\text{act}}^e} \frac{\lambda_m^e}{H} \sum_{k=0}^{H-1} \|\hat{\mathbf{a}}_{t+k}^{e,m} - \mathbf{a}_{t+k}^{e,m}\|_1, \quad (10)$$

where λ_m^e weights action modality m . Because the sum runs over $\mathcal{M}_{\text{act}}^e$, action modalities that e does not provide are excluded. In both Eq. (10) and Eq. (8), chunk steps and dreaming offsets beyond the end of the trajectory are padded and excluded from the average. The objective for embodiment e on a dataset $\mathcal{D}$ is

$$\mathcal{L}^e(\mathcal{D}) = \mathbb{E}_{\mathcal{D}} [\mathcal{L}_{\text{act}}^e + \lambda_{\text{dream}}^e \mathcal{L}_{\text{dream}}^e], \quad (11)$$

where $\mathbb{E}_{\mathcal{D}}$ averages over time steps t of all trajectories in $\mathcal{D}$, and $\lambda_{\text{dream}}^e \geq 0$ weights embodiment dreaming.

Pretraining. Let $\mathcal{E}_{\text{pre}} \subseteq \mathcal{E}$ be the pretraining embodiments, including humans and multiple robots. Each batch mixes samples from one or more embodiment datasets, and MXT+ minimizes

$$\mathcal{L}_{\text{pre}} = \sum_{e \in \mathcal{E}_{\text{pre}}} \mathcal{L}^e(\mathcal{D}_e). \quad (12)$$

The shared trunk G_{θ} receives gradients from all embodiments, whereas the embodiment-specific modules of e are updated only by $\mathcal{D}_e$.

Fine-tuning. Given a target embodiment e^* and a target-task dataset $\mathcal{D}_{e^*}^{\text{task}}$, EgoX initializes MXT+ from the pretrained checkpoint and minimizes

$$\mathcal{L}_{\text{ft}} = \mathcal{L}^{e^*}(\mathcal{D}_{e^*}^{\text{task}}), \quad (13)$$

where setting $\lambda_{\text{dream}}^{e^*} = 0$ disables dreaming during fine-tuning. When the target task has little data, the pretrained trunk provides a manipulation prior learned from human and other-robot demonstrations, and the embodiment-specific modules adapt the shared representation to the target robot’s sensors and action interface. Unlike single-source human-to-robot pretraining, EgoX does not rely on a single source or target embodiment; it co-trains on a broad embodiment mixture and then specializes to the target deployment.

Inference. At deployment, MXT+ retains the modality tokenizers and student embodiment tokenizer $\phi_{\text{dream}}^{e^*}$. The latter aggregates current observation features into embodiment tokens, which are concatenated with the modality tokens and passed to the shared trunk. The action experts decode action chunks $\hat{A}_t^{e^*}$ for low-level control. The dreaming expert $\psi_{\text{dream}}^{e^*}$ and EMA teachers are used only during training; inference requires neither future-latent prediction nor EMA target computation. Algorithm 1 in Appendix A.4 summarizes the training procedure.

4 EXPERIMENTS AND RESULTS

4.1 EXPERIMENTAL SETUP

Tasks. We evaluate EgoX on five diverse manipulation tasks covering long-horizon locomotive manipulation, semantic object selection, precise alignment, tool use, and contact-rich interaction. In *Kitchen-Bread* (Sauce Spreading), the robot retrieves sauce containers from a fridge or tabletop, picks up a spreading tool, dips it into the sauce, and spreads the sauce onto bread. In *Kitchen-Cleanup* (Tool Organization), the robot picks up tools from a tabletop and places them into a container, with the tools and container initialized around a plate fixed at the center of a designated tabletop region. In *Sort* (Block Sorting), the robot sorts blocks into containers of the corresponding colors. In *Insert* (T-shaped Block Insertion), the robot picks up a T-shaped block, aligns it with its base, and inserts it. In *Scoop* (Cat Litter Scooping), the robot uses a grasped or fixed scoop to collect cat litter positioned inside a litter box, empties it into a bin, and returns the scoop to the edge of the litter box. Figs. 11–15 in the appendix show example rollouts of each task across the evaluated embodiments.

Embodiments. Our evaluation covers six robot embodiments (five physical robots and one simulated humanoid) that differ in morphologies, mobility, handedness, and end-effectors. *G1-Dex5* is a bimanual Unitree G1 humanoid equipped with five-finger Inspire RH56F1 dexterous hands. *xArm7-Dex4* is a tabletop xArm7 equipped with a four-finger LEAP Hand (Shaw et al., 2023); *G1-Sim-Dex3* is a simulated Unitree G1 equipped with three-finger Dex3-1 dexterous hands. *LocoMan-Dex2* (Lin et al., 2024) is a quadruped with two-finger grippers mounted on its front legs, evaluated in unimanual mode. *RLite-Gripper* is a Galaxea R1 Lite wheeled bimanual mobile manipulator with parallel-jaw grippers. *Z1-Tool* is a tabletop Unitree Z1 arm with

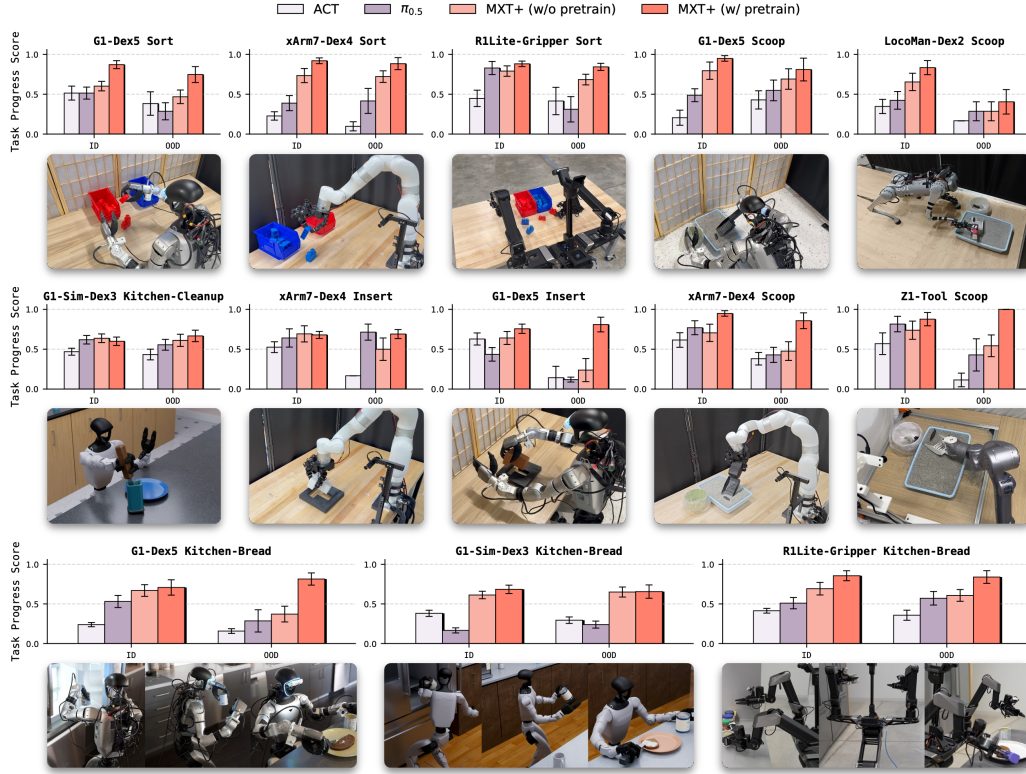


Figure 3: **Main results.** Normalized task progress score of ACT, $\pi_{0.5}$, MXT+ without pretraining, and MXT+ with cross-embodiment pretraining on 13 task-embodiment pairs spanning five tasks (Kitchen-Bread, Kitchen-Cleanup, Sort, Insert, Scoop) and six embodiments (five physical robots and the simulated G1-Sim-Dex3). Each panel reports ID and OOD performance.

a fixed, non-actuated scoop. Egocentric human demonstrations (Human-Dex5) provide an additional embodiment for cross-embodiment pretraining. Together, these platforms span dexterous hands, grippers, and fixed tools across diverse robot morphologies and mobility configurations.

Baselines and training settings. We compare MXT+ with ACT (Zhao et al., 2023) and $\pi_{0.5}$ (Black et al., 2025a), and evaluate MXT+ both with and without cross-embodiment pretraining. ACT is trained directly on demonstrations from the target embodiment and task. *MXT+ without pretraining* is likewise trained from scratch on target-embodiment demonstrations, using the same modular tokenizers, shared Transformer trunk, and action experts as the pretrained variant. *MXT+ with pretraining* is first trained on a mixture of egocentric demonstrations from humans, physical robots, and simulated robots, and then fine-tuned on the target embodiment and task following Section 3.4. The comparison between these two MXT+ variants evaluates the benefit of cross-embodiment pretraining without changing the downstream policy architecture, while ACT and $\pi_{0.5}$ serve as baselines (Appendix A.5) representing Transformer-based imitation learning and pretrained VLA models fine-tuned on target-embodiment demonstrations, respectively. We additionally study the sources of transfer through pretraining-mixture and model ablations. Specifically, we vary the embodiments included in pretraining to examine the effects of embodiment diversity and composition, compare models with and without embodiment dreaming and assess the contribution of the modular design.

Evaluation settings and metrics. For each task and embodiment, we assess policies under in-distribution (ID) and out-of-distribution (OOD) conditions, defined relative to the target-embodiment training demonstrations. ID settings use objects, initial configurations, and environmental conditions similar to those encountered during target training, while OOD settings introduce task-specific held-out variations, such as novel object instances, appearances, poses, or scene configurations. Each real-world task-embodiment pair is evaluated over 20 trials (13 ID and 7 OOD), and each simulated pair over 50 trials (32 ID and 18 OOD). We report the *average normalized*

task progress score, which credits partial completion of intermediate task stages, adds a one-point success bonus, and normalizes the accumulated progress by the task’s maximum score.

4.2 EXPERIMENTAL EVALUATION

Our evaluation aims to address the following questions: (1) Does *egocentric cross-embodiment pre-training* improve downstream dexterous manipulation compared to training from scratch on target-embodiment data? (2) How do the *diversity and composition* of pretraining embodiments affect downstream performance? (3) Does *embodiment dreaming* improve cross-embodiment learning? (4) Does Latent Geometry Become More Similar Across Embodiments? (5) Does *modular design* enhance the cross-embodiment transfer?

Does egocentric cross-embodiment pretraining outperform training from scratch? We examine whether egocentric cross-embodiment pretraining improves downstream policy learning compared with training the same policy architecture from scratch on target-embodiment demonstrations. In Fig. 3, cross-embodiment pretraining improves the normalized task progress score of MXT+ on 11 of 13 task-embodiment pairs under ID evaluation and all 13 pairs under OOD evaluation.

Under ID evaluation, cross-embodiment pretraining yields particularly large gains on G1-Dex5 Sort, LocoMan-Dex2 Scoop, and xArm7-Dex4 Scoop. The benefit becomes even more pronounced under OOD evaluation, with substantial improvements on G1-Dex5 Kitchen-Bread, G1-Dex5 Insert, and Z1-Tool Scoop, where pretrained MXT+ substantially closes the gap to or exceeds the performance of other baselines.

Overall, these results suggest that cross-embodiment pretraining on mixed human-robot demonstrations exposes the policy to a broader range of visual, geometric, and interaction patterns than target-task demonstrations alone, contributing to improved downstream generalization.

Does pretraining benefit from embodiment diversity? We next study how the composition of the pretraining mixture affects downstream performance. As shown in Fig. 4, we compare four settings: no pretraining, human-only pretraining, human plus other-robot pretraining, and full cross-embodiment pretraining. Human-only pretraining uses egocentric human demonstrations; human plus other-robot pretraining additionally incorporates demonstrations from robot embodiments other than the target robot; and full cross-embodiment pretraining further includes demonstrations from the target embodiment on non-target tasks. All variants are fine-tuned on the same target-embodiment, target-task demonstrations. We evaluate this ablation across both real-robot and simulation settings, covering Sort on R1Lite-Gripper, Sort and Insert on G1-Dex5 and xArm7-Dex4, as well as Kitchen-Bread on simulated G1-Sim-Dex3.

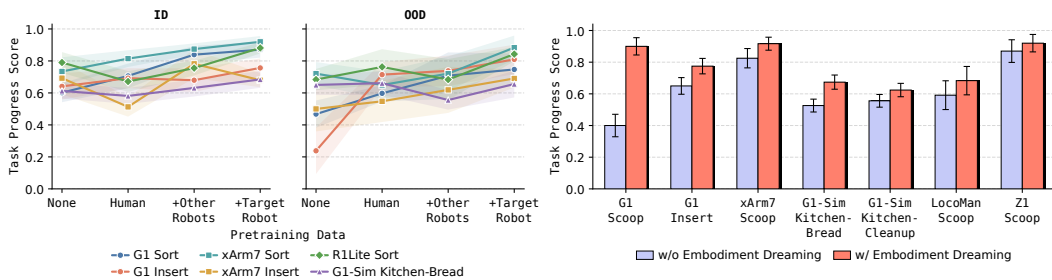


Figure 4: **Pretraining data composition and embodiment dreaming ablations.** *Left and middle:* ID and OOD task progress scores as pretraining data is added: none, human data, then other robots, then the target robot’s other tasks (full cross-embodiment). *Right:* task progress scores with and without embodiment dreaming. Labels omit end effectors. Shading and error bars: ± 1 SEM.

Every pretraining mixture improves average OOD performance over no pretraining, and the gains grow as the mixture broadens. Human-only pretraining already improves OOD performance on five of six pairs, most strongly on G1-Dex5 Insert, suggesting that egocentric human demonstrations can provide transferable manipulation priors despite the embodiment gap. Adding other robots’ demonstrations raises ID performance on five of six pairs, and adding the target robot’s demonstrations on other tasks improves OOD performance on every pair. Full cross-embodiment pretraining performs best, improving the average task progress score over no pretraining by 0.12 on ID and 0.23 on OOD, which highlights the value of broad and diverse pretraining data.

Does embodiment dreaming improve cross-embodiment learning? Finally, we study the effect of embodiment dreaming by comparing MXT+ with MXT, its counterpart without the embodiment-dreaming objective. We evaluate this ablation across five embodiments and seven task-embodiment settings, including Scoop on four embodiments (G1-Dex5, xArm7-Dex4, LocoMan-Dex2, and Z1-Tool), Insert on G1-Dex5, and both Kitchen-Bread and Kitchen-Cleanup on G1-Sim-Dex3. The gains are observed across different tasks and embodiments, with particularly pronounced improvements on G1-Dex5, suggesting that embodiment dreaming provides an additional learning signal beyond action prediction, improving downstream manipulation performance. These results provide evidence that embodiment dreaming can improve representations for cross-embodiment learning, facilitating knowledge transfer across substantially different kinematics, viewpoints, and action spaces.

Does Latent Geometry Become More Similar Across Embodiments? We compute Grovov-Wasserstein (GW) distances between embodiment-specific distributions of shared-trunk latent representations extracted from Scoop demonstrations.

Each embodiment’s demonstrations are encoded by its corresponding checkpoint, with matched samples across conditions. GW uses median-normalized within-distribution Euclidean distances in the original latent space. We report percentage changes in GW relative to each baseline, with negative values indicating greater geometric similarity. We compare MXT+ with and without cross-embodiment pretraining. To assess embodiment dreaming, we compare MXT with MXT+, both pretrained and fine-tuned. Fig. 5 shows that pretraining lowers GW for 9 of 10 embodiment pairs and reduces mean GW across the pairs by 21.30%. MXT+ yields lower GW than MXT for all 10 pairs, reducing mean GW across the pairs by 38.52%. These findings suggest more similar internal distance structures among embodiment-specific latent representations for the same manipulation task.

	G1-Dex5	xArm7-Dex4	Z1-Tool	LocoMan-Dex2	Human-Dex5
G1-Dex5		-24.8%	-46.6%	-34.5%	-22.5%
xArm7-Dex4	-18.3%		-29.7%	-31.9%	-30.8%
Z1-Tool	-6.2%	-8.0%		-55.6%	-59.9%
LocoMan-Dex2	-9.7%	-8.3%	+31.7%		-36.7%
Human-Dex5	-32.3%	-31.2%	-42.3%	-38.5%	

Figure 5: GW change across embodiment pairs. **Pretraining:** red, lower triangle; **Embodiment dreaming:** blue, upper triangle.

Does modular design enhance the cross-embodiment transfer? We next examine the role of our modular architecture by comparing MXT+ with a coarse variant that merges modality-specific inputs and actions. Fig. 6 evaluates both variants with and without cross-embodiment pretraining on Z1-Tool Scoop, xArm7-Dex4 Scoop, G1-Sim-Dex3 Kitchen-Bread, and G1-Sim-Dex3 Kitchen-Cleanup. The modular design consistently outperforms the coarse variant, with particularly pronounced gains for the higher-dimensional xArm7-Dex4 and G1-Sim-Dex3, indicating that modality-specific interfaces benefit both downstream learning and cross-embodiment transfer.

The benefit of modularity is especially clear on Kitchen-Cleanup: after pretraining, the coarse variant achieves zero task progress, while modular MXT+ retains substantial performance. The first step of this task requires a 3D velocity command for locomotion, a small but critical modality within G1’s 51-dimensional action space. Treating all actions as a single coarse representation can hinder learning this modality, whereas the modular design provides dedicated experts for each action modality. Overall, these results show that

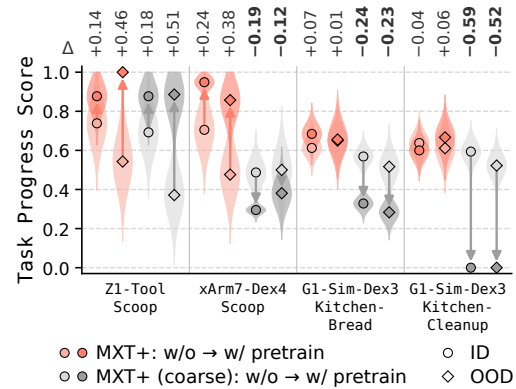


Figure 6: **Coarse ablation.** Arrows go from w/o to w/ pretraining; violins show each mean’s uncertainty (from its SEM).

modularity is crucial for both downstream performance and effective cross-embodiment transfer, particularly for high-dimensional embodiments with diverse modalities.

5 CONCLUSION

We presented EgoX, an egocentric cross-embodiment framework that represents human, real-robot, and simulated demonstrations in a unified, modality-structured format and trains MXT+, a Modularized Cross-Embodiment Transformer with Embodiment Dreaming. Embodiment-specific tokenizers and action experts preserve each embodiment’s sensing and control interfaces around a shared trunk, and embodiment dreaming trains this trunk to predict future embodiment latents, modeling each body’s dynamics at no inference cost. Across five tasks and six embodiments, cross-embodiment pretraining improves downstream manipulation, broader embodiment mixtures improve transfer, and embodiment dreaming yields further gains, especially on contact-rich tool use.

REFERENCES

- Joel A. E. Andersson, Joris Gillis, Greg Horn, James B. Rawlings, and Moritz Diehl. CasADi: A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.
- Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.
- Robert Baines, Frank Fish, Josh Bongard, and Rebecca Kramer-Bottiglio. Robots that evolve on demand. *Nature Reviews Materials*, 9:822–835, 2024. doi: 10.1038/s41578-024-00711-z. URL <https://www.nature.com/articles/s41578-024-00711-z>.
- Hongzhe Bi, Lingxuan Wu, Tianwei Lin, Hengkai Tan, Zhizhong Su, Hang Su, and Jun Zhu. H-RDT: Human manipulation enhanced bimanual robotic manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18135–18143, 2026. doi: 10.1609/aaai.v40i22.38875. URL <https://ojs.aaai.org/index.php/AAAI/article/view/38875>.
- Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *Proceedings of The 9th Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 2025a.
- Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0 : A vision-language-action flow model for general robot control. In *Robotics: Science and Systems (RSS)*, 2025b.
- Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025c.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 1877–1901, 2020.
- Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Learning to act anywhere with task-centric latent actions. In *Robotics: Science and Systems (RSS)*, 2025. doi: 10.15607/RSS.2025.XXI.014. URL <https://www.roboticsproceedings.org/rss21/p014.html>. arXiv:2505.06111.
- Justin Carpentier, Guilhem Saurel, Gabriele Buondonno, Joseph Mirabel, Florent Lamiraux, Olivier Stasse, and Nicolas Mansard. The Pinocchio C++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In *IEEE/SICE International Symposium on System Integration (SII)*, 2019.

- Lawrence Yunliang Chen, Karthik Dharmarajan, Kush Hari, Chenfeng Xu, Quan Vuong, and Ken Goldberg. MIRAGE: Cross-embodiment zero-shot policy transfer with cross-painting. In *Robotics: Science and Systems (RSS)*, 2024.
- Sirui Chen, Chen Wang, Kaden Nguyen, Li Fei-Fei, and C Karen Liu. ARCap: Collecting High-quality Human Demonstrations for Robot Learning with Augmented Reality Feedback. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8291–8298. IEEE, 2025.
- Xuxin Cheng, Jialong Li, Shiqi Yang, Ge Yang, and Xiaolong Wang. Open-TeleVision: Teleoperation with immersive active visual feedback. In *Conference on Robot Learning (CoRL)*, 2024.
- Abhay Deshpande, Maya Guru, Rose Hendrix, Snehal Jauhri, Ainaz Eftekhari, Rohun Tripathi, Max Argus, Jordi Salvador, Haoquan Fang, Matthew Wallingford, et al. MolmoB0T: Large-scale simulation enables zero-shot manipulation. *arXiv preprint arXiv:2603.16861*, 2026.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 4171–4186, 2019.
- Ria Doshi, Homer Walke, Oier Mees, Sudeep Dasari, and Sergey Levine. Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation. In *Proceedings of The 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 496–512. PMLR, 2025. URL <https://proceedings.mlr.press/v270/doshi25a.html>.
- Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.
- Yankai Fu, Ning Chen, Junkai Zhao, Shaozhe Shan, Guocai Yao, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. METIS: Multi-Source Egocentric Training for Integrated Dexterous Vision-Language-Action Model. *arXiv preprint arXiv:2511.17366*, 2025. doi: 10.48550/arXiv.2511.17366. URL <http://arxiv.org/abs/2511.17366>.
- Zipeng Fu, Qingqing Zhao, Qi Wu, Gordon Wetzstein, and Chelsea Finn. HumanPlus: Humanoid shadowing and imitation from humans. In *Proceedings of The 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2828–2844. PMLR, 2024.
- Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, et al. DreamDojo: A real-time robot world model from large-scale human videos. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026.
- Gemini Robotics Team, Abbas Abdolmaleki, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Ashwin Balakrishna, Nathan Batchelor, Alex Bewley, Jeff Bingham, et al. Gemini Robotics 1.5: Pushing the frontier of generalist robots with advanced embodied reasoning, thinking, and motion transfer. *arXiv preprint arXiv:2510.03342*, 2025a.
- Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini Robotics: Bringing AI into the physical world. *arXiv preprint arXiv:2503.20020*, 2025b.
- Raktim Gautam Goswami, Amir Bar, David Fan, Tsung-Yen Yang, Gaoyue Zhou, Prashanth Krishnamurthy, Michael Rabbat, Farshad Khorrami, and Yann LeCun. World Models for Learning Dexterous Hand-Object Interactions from Human Videos. *arXiv preprint arXiv:2512.13644*, 2025. doi: 10.48550/arXiv.2512.13644. URL <http://arxiv.org/abs/2512.13644>.
- Ankur Handa, Karl Van Wyk, Wei Yang, Jacky Liang, Yu-Wei Chao, Qian Wan, Stan Birchfield, Nathan Ratliff, and Dieter Fox. DexPilot: Vision-based teleoperation of dexterous robotic hand-arm system. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

- Ryan Hoque, Peide Huang, David J Yoon, Mouli Sivapurapu, and Jian Zhang. EgoDex: Learning dexterous manipulation from large-scale egocentric video. In *International Conference on Learning Representations (ICLR)*, 2026.
- Simar Kareer, Dhruv Patel, Ryan Punamiya, Pranay Mathur, Shuo Cheng, Chen Wang, Judy Hoffman, and Danfei Xu. EgoMimic: Scaling imitation learning via egocentric video. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13226–13233. IEEE, 2025.
- Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, et al. OpenVLA: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025. URL <https://proceedings.mlr.press/v270/kim25c.html>.
- Robin Jeanne Kirschner, Kübra Karacan, Alessandro Melone, and Sami Haddadin. Categorizing robots by performance fitness into the tree of robots. *Nature Machine Intelligence*, 7(3):459–470, 2025. doi: 10.1038/s42256-025-00995-y. URL <https://www.nature.com/articles/s42256-025-00995-y>.
- Marion Lepert, Ria Doshi, and Jeannette Bohg. SHADOW: Leveraging segmentation masks for cross-embodiment policy transfer. In *Proceedings of The 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3536–3550. PMLR, 2025. URL <https://proceedings.mlr.press/v270/lepert25a.html>.
- Baoyu Li, Xinchun Yin, Mengying Lin, Yixin Zhang, and Danfei Xu. EgoWAM: World Action Models Beyond Pixels with In-the-Wild Egocentric Human Data. *arXiv preprint arXiv:2607.08436*, 2026a. doi: 10.48550/arXiv.2607.08436. URL <https://arxiv.org/abs/2607.08436>.
- Boyu Li, Chaoyi Xu, Haoqi Yuan, Xinrun Xu, Börje F. Karlsson, Dongbin Zhao, Haoran Li, and Zongqing Lu. X-DiffVLA: X-embodied diffusion action heads for vision-language-action models. In *Robotics: Science and Systems (RSS)*, 2026b.
- Changyi Lin, Xingyu Liu, Yuxiang Yang, Yaru Niu, Wenhao Yu, Tingnan Zhang, Jie Tan, Byron Boots, and Ding Zhao. LocoMan: Advancing versatile quadrupedal dexterity with lightweight loco-manipulators. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6877–6884. IEEE, 2024.
- Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation. In *International Conference on Learning Representations (ICLR)*, 2025.
- Fanqi Lin, Kushal Arora, Jean Mercat, Haruki Nishimura, Paarth Shah, Chen Xu, Mengchao Zhang, Mark Zolotas, Maya Angeles, Owen Pfannenstiehl, Andrew Beaulieu, and Jose Barreiros. A systematic study of data modalities and strategies for co-training large behavior models for robot manipulation. In *Robotics: Science and Systems (RSS)*, 2026a.
- Kevin Lin, Ajay Mandlekar, Caelan Reed Garrett, Nikita Chernyadev, Yu Fang, Runyu Ding, Yuqi Xie, Justin Tran, Linxi Fan, and Yuke Zhu. HumanoidMimicGen: Data generation for loco-manipulation via whole-body planning. *arXiv preprint arXiv:2605.27724*, 2026b.
- Yunfan Lou, Yifan Ye, Yankai Fu, Jun Cen, Xiaowei Chi, Yaoyu Lyu, Peidong Jia, Sirui Han, Zhihe Lu, and Shanghang Zhang. Dream-Tac: A unified tactile world action model for contact-rich robot manipulation. *arXiv preprint arXiv:2606.08737*, 2026.
- Hao Luo, Ye Wang, Wanpeng Zhang, Sipeng Zheng, Ziheng Xi, Chaoyi Xu, Haiweng Xu, Haoqi Yuan, Chi Zhang, Yiqing Wang, Yicheng Feng, and Zongqing Lu. Being-H0.5: Scaling human-centric robot learning for cross-embodiment generalization. *arXiv preprint arXiv:2601.12993*, 2026a. URL <https://arxiv.org/abs/2601.12993>.
- Hao Luo, Wanpeng Zhang, Yicheng Feng, Sipeng Zheng, Haiweng Xu, Chaoyi Xu, Ziheng Xi, Yuhui Fu, and Zongqing Lu. Being-H0.7: A latent world-action model from egocentric videos. *arXiv preprint arXiv:2605.00078*, 2026b.

- Abhiram Maddukuri, Zhenyu Jiang, Lawrence Yunliang Chen, Soroush Nasiriany, Yuqi Xie, Yu Fang, Wenqi Huang, Zu Wang, Zhenjia Xu, Nikita Chernyadev, Scott Reed, Ken Goldberg, Ajay Mandlekar, Linxi Fan, and Yuke Zhu. Sim-and-real co-training: A simple recipe for vision-based robotic manipulation. In *Robotics: Science and Systems (RSS)*, 2025.
- Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A universal visual representation for robot manipulation. In *Proceedings of The 6th Conference on Robot Learning (CoRL)*, volume 205 of *Proceedings of Machine Learning Research*, pp. 892–909. PMLR, 2023. URL <https://proceedings.mlr.press/v205/nair23a.html>.
- Yaru Niu, Yunzhe Zhang, Mingyang Yu, Changyi Lin, Chenhao Li, Yikai Wang, Yuxiang Yang, Wenhao Yu, Tingnan Zhang, Zhenzhen Li, Jonathan Francis, Bingqing Chen, Jie Tan, and Ding Zhao. Human2LocoMan: Learning versatile quadrupedal manipulation with human pretraining. In *Robotics: Science and Systems (RSS)*, 2025.
- Yaru Niu, Zhenlong Fang, Binghong Chen, Shuai Zhou, Revanth Krishna Senthilkumaran, Hao Zhang, Bingqing Chen, Chen Qiu, H Eric Tseng, Jonathan Francis, and Ding Zhao. Learning versatile humanoid manipulation with touch dreaming. *arXiv preprint arXiv:2604.13015*, 2026.
- NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.
- Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. In *Robotics: Science and Systems (RSS)*, 2024.
- Abby O’Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open X-Embodiment: Robotic learning datasets and RT-X models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.
- Younghyo Park and Pulkit Agrawal. Using Apple Vision Pro to train and control robots. <https://github.com/Improbable-AI/VisionProTeleop>, 2024.
- Austin Patel and Shuran Song. GET-Zero: Graph embodiment transformer for zero-shot embodiment generalization. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14262–14269, 2025.
- Rolf Pfeifer, Max Lungarella, and Fumiya Iida. Self-organization, embodiment, and biologically inspired robotics. *Science*, 318(5853):1088–1093, 2007. doi: 10.1126/science.1145803. URL <https://www.science.org/doi/10.1126/science.1145803>.
- Ryan Punamiya, Dhruv Patel, Patcharapong Aphiwetsa, Pranav Kuppili, Lawrence Y. Zhu, Simar Kareer, Judy Hoffman, and Danfei Xu. EgoBridge: Domain adaptation for generalizable imitation from egocentric human data. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pp. 47092–47122, 2025. doi: 10.52202/085713-1569. URL https://papers.neurips.cc/paper_files/paper/2025/hash/434756a76c44e220d5f32d6e0b488c0c-Abstract-Conference.html.
- Ryan Punamiya, Simar Kareer, Zeyi Liu, Josh Citron, Ri-Zhao Qiu, Xiongyi Cai, Alexey Gavryushin, Jiaqi Chen, Davide Liconti, Lawrence Y Zhu, et al. EgoVerse: An egocentric human dataset for robot learning from around the world. In *Robotics: Science and Systems (RSS)*, 2026.
- Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dieter Fox. AnyTeleop: A general vision-based dexterous robot arm-hand teleoperation system. In *Robotics: Science and Systems (RSS)*, 2023.
- Ri-Zhao Qiu, Shiqi Yang, Xuxin Cheng, Chaitanya Chawla, Jialong Li, Tairan He, Ge Yan, David J Yoon, Ryan Hoque, Lars Paulsen, et al. Humanoid policy $\sim$ human policy. In *Proceedings of The 9th Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 2888–2906. PMLR, 2025.

- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021.
- Daniel Rakita, Bilge Mutlu, and Michael Gleicher. RelaxedIK: Real-time synthesis of accurate and feasible robot arm motion. In *Robotics: Science and Systems (RSS)*, 2018.
- Carmelo Sferrazza, Dun-Ming Huang, Fangchen Liu, Jongmin Lee, and Pieter Abbeel. Body transformer: Leveraging robot embodiment for policy learning. In *Proceedings of The 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3407–3424. PMLR, 2025. URL <https://proceedings.mlr.press/v270/sferrazza25a.html>.
- Kenneth Shaw, Ananye Agarwal, and Deepak Pathak. LEAP Hand: Low-cost, efficient, and anthropomorphic hand for robot learning. In *Robotics: Science and Systems (RSS)*, 2023.
- Modi Shi, Shijia Peng, Jin Chen, Haoran Jiang, Tianyu Li, Di Huang, Ping Luo, Hongyang Li, and Li Chen. Unlocking in-the-wild loco-manipulation with robot-free egocentric demonstration. In *Robotics: Science and Systems (RSS)*, 2026. doi: 10.15607/RSS.2026.XXII.204. URL <https://www.roboticsproceedings.org/rss22/p204.html>.
- Mohan Kumar Srirama, Sudeep Dasari, Shikhar Bahl, and Abhinav Gupta. HRP: Human affordances for robotic pre-training. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.
- TRI LBM Team, Jose Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-Hsin Fang, Kunimatsu Hashimoto, Muhammad Zubair Irshad, et al. A careful examination of large behavior models for multitask dexterous manipulation. *arXiv preprint arXiv:2507.05331*, 2025.
- Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1): 25–57, 2006.
- Lirui Wang, Xinlei Chen, Jialiang Zhao, and Kaiming He. Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 124420–124450, 2024.
- Songlin Wei, Hongyi Jing, Boqian Li, Zhenyu Zhao, Jiageng Mao, Zhenhao Ni, Sicheng He, Jie Liu, Xiawei Liu, Kaidi Kang, Sheng Zang, Weiduo Yuan, Marco Pavone, Di Huang, and Yue Wang. Ψ_0 : An open foundation model towards universal humanoid loco-manipulation. In *Robotics: Science and Systems (RSS)*, 2026a. doi: 10.15607/RSS.2026.XXII.021. URL <https://www.roboticsproceedings.org/rss22/p021.html>.
- Zhenyu Wei, Yunchao Yao, and Mingyu Ding. One hand to rule them all: Canonical representations for unified dexterous manipulation. In *Robotics: Science and Systems (RSS)*, 2026b.
- Xiaomeng Xu, Jisang Park, Han Zhang, Eric Cousineau, Aditya Bhat, Jose Barreiros, Dian Wang, Jeannette Bohg, and Shuran Song. HoMMI: Learning whole-body mobile manipulation from human demonstrations. In *Robotics: Science and Systems (RSS)*, 2026.
- Ruihan Yang, Qinxi Yu, Yecheng Wu, Rui Yan, Borui Li, An-Chieh Cheng, Xueyan Zou, Yunhao Fang, Xuxin Cheng, Ri-Zhao Qiu, Hongxu Yin, Sifei Liu, Song Han, Yao Lu, and Xiaolong Wang. EgoVLA: Learning Vision-Language-Action Models from Egocentric Human Videos. *arXiv preprint arXiv:2507.12440*, 2025. URL <https://arxiv.org/abs/2507.12440>.
- Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, et al. Latent action pretraining from videos. In *International Conference on Learning Representations (ICLR)*, 2025.

- Chengbo Yuan, Rui Zhou, Mengzhen Liu, Yingdong Hu, Shengjie Wang, Li Yi, Chuan Wen, Shanghang Zhang, and Yang Gao. MotionTrans: Human VR Data Enable Motion-Level Learning for Robotic Manipulation Policies. *arXiv preprint arXiv:2509.17759*, 2025. URL <https://arxiv.org/abs/2509.17759>.
- Chubin Zhang, Jianan Wang, Zifeng Gao, Yue Su, Tianru Dai, Cai Zhou, Jiwen Lu, and Yansong Tang. CLAP: Contrastive Latent Action Pretraining for Learning Vision-Language-Action Models from Human Videos. *arXiv preprint arXiv:2601.04061*, 2026. doi: 10.48550/arXiv.2601.04061. URL <http://arxiv.org/abs/2601.04061>.
- Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems (RSS)*, 2023.
- Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loïc Magne, et al. FLARE: Robot learning with implicit world modeling. In *Proceedings of The 9th Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 3952–3971. PMLR, 2025. URL <https://proceedings.mlr.press/v305/zheng25a.html>.
- Ruijie Zheng, Dantong Niu, Yuqi Xie, Jing Wang, Mengda Xu, Yunfan Jiang, Fernando Castañeda, Fengyuan Hu, You Liang Tan, Letian Fu, et al. EgoScale: Scaling dexterous manipulation with diverse egocentric human data. *arXiv preprint arXiv:2602.16710*, 2026.
- Lawrence Y. Zhu, Pranav Kuppili, Ryan Punamiya, Patcharapong Aphiwetsa, Dhruv Patel, Simar Kareer, Sehoon Ha, and Danfei Xu. EMMA: Scaling Mobile Manipulation via Egocentric Human Data. *IEEE Robotics and Automation Letters*, 11(3):3087–3094, 2026.

A APPENDIX

A.1 HANDLING MISSING MODALITIES

Not every embodiment provides every modality in the shared vocabulary $\mathcal{M}$ (Sec. 3.1); for example, single-arm embodiments have no left wrist or left end-effector link. EgoX handles such gaps so that token sequences remain semantically aligned across embodiments. For each embodiment e , the modality indicator

$$\delta_m^e = \mathbb{I}[m \text{ is available for } e] \in \{0, 1\} \quad (14)$$

defines the available observation modalities $\mathcal{M}_{\text{obs}}^e = \{m \in \mathcal{M}_{\text{obs}} : \delta_m^e = 1\}$, and $\mathcal{M}_{\text{act}}^e$ is defined analogously. Unavailable observation modalities are replaced by learnable mask tokens (Appendix A.2), so the shared trunk always receives the same modality layout, and unavailable action modalities are excluded from the action loss (Sec. 3.4).

A.2 DETAILS ON MXT+

Modality tokenizers. Each tokenizer ϕ_m^e maps modality m to a fixed number L_m of d -dimensional tokens. Modalities are defined at an anatomical or functional level, e.g., egocentric vision, body pose, end-effector pose, end-effector joint pose, and end-effector keypoints, so embodiments with different morphologies expose compatible interfaces whenever they share a modality. Image modalities are encoded by a visual backbone f_{vis} shared across all embodiments, followed by an embodiment-specific cross-attention aggregator g_m^e , so that $\phi_m^e = g_m^e \circ f_{\text{vis}}$. State modalities are encoded by lightweight MLPs followed by cross-attention aggregation. Unavailable modalities are replaced by mask tokens (Appendix A.1), and the tokens of all modalities are concatenated in a fixed modality order:

$$\mathbf{Z}_t^e = [\mathbf{Z}_t^{e,m}]_{m \in \mathcal{M}_{\text{obs}}^e} \in \mathbb{R}^{L \times d}, \quad L = \sum_{m \in \mathcal{M}_{\text{obs}}^e} L_m. \quad (15)$$

Tokenizing per modality preserves each modality’s semantic identity and avoids mixing heterogeneous signals too early. The trunk therefore always consumes the same token interface, while the tokenizers absorb embodiment-specific differences in dimensionality, camera availability, and kinematic parameterization.

Shared Transformer trunk. The trunk G_θ is an encoder-decoder Transformer. The encoder applies self-attention over $\mathbf{Z}_t^e$, and the decoder cross-attends from N_q learnable query tokens $\mathbf{Q} \in \mathbb{R}^{N_q \times d}$ to the encoder outputs:

$$\mathbf{U}_t^e = G_\theta(\mathbf{Z}_t^e; \mathbf{Q}) \in \mathbb{R}^{N_q \times d}. \quad (16)$$

Because every embodiment is presented through the same token interface, G_θ is trained on all embodiments without requiring their raw observations or actions to share a dimensionality. The trunk is therefore the main conduit for cross-embodiment transfer, while the tokenizers and experts retain the embodiment-specific details needed for execution.

Modular action experts. For each available action modality $m \in \mathcal{M}_{\text{act}}^e$, the action expert decodes the corresponding action chunk from the trunk output:

$$\hat{\mathbf{A}}_t^{e,m} = \psi_m^e(\mathbf{U}_t^e) \in \mathbb{R}^{H \times d_m}. \quad (17)$$

Action modalities are drawn from the shared vocabulary $\mathcal{M}_{\text{act}}$, so anatomically or functionally corresponding outputs occupy the same modality across embodiments. For example, the end-effector pose of a single-arm robot corresponds to the right end-effector pose of a bimanual embodiment. The experts remain embodiment-specific to handle differences in dimensionality, control semantics, and validity masks, while the structure shared across embodiments is learned by the trunk.

A.3 SUCCESS RATES

Sec. 4 reports the normalized task progress score P . Fig. 7 reports the success rate on the same trials, i.e., the fraction of trials with $b = 1$ (Appendix A.10), for each method and task–embodiment pair under ID and OOD conditions. Each real-world pair is evaluated over 20 trials (13 ID and 7 OOD), and each simulated pair over 50 trials (32 ID and 18 OOD).

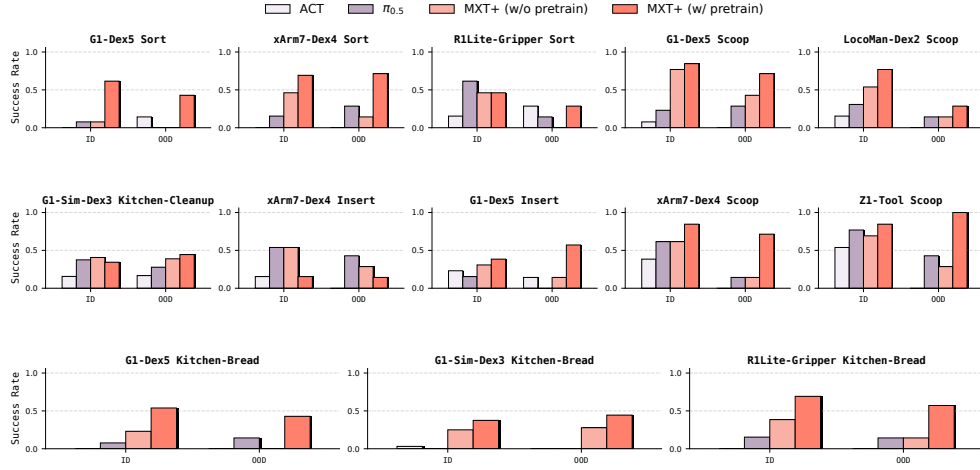


Figure 7: **Success rates.** Success rate of ACT, $\pi_{0.5}$, MXT+ without pretraining, and MXT+ with cross-embodiment pretraining on each task–embodiment pair under ID and OOD conditions, evaluated on the same trials as Fig. 3.

A.4 TRAINING ALGORITHM

Algorithm 1 details the full EgoX training procedure, including cross-embodiment pretraining, target fine-tuning, embodiment dreaming, and the EMA target update. The rest of this section describes each step, and Table 1 lists the hyperparameters.

Algorithm 1 EgoX pretraining and fine-tuning with embodiment dreaming

Input: Datasets $\{\mathcal{D}_e\}_{e \in \mathcal{E}_{\text{pre}}}$, target dataset $\mathcal{D}_{e^*}^{\text{task}}$
Input: Action horizon H , dreaming offsets $\mathcal{K}$, EMA decay α , per-embodiment batch size B
Output: Target policy $\pi^{e^*} = \Psi^{e^*} \circ G_\theta \circ \Phi^{e^*}$

- 1: Initialize tokenizers, trunk G_θ , action experts, and dreaming experts
- 2: Initialize each teacher $\tilde{\phi}$ as a copy of its student ϕ
- 3: **for** each pretraining step **do**
- 4: **for** each embodiment $e \in \mathcal{E}_{\text{pre}}$ **do**
- 5: Sample B time steps t from $\mathcal{D}_e$, with observations $\mathbf{o}_t^e$, action chunks, and future observations $\mathbf{o}_{t+k}^e$ for $k \in \mathcal{K}$
- 6: Predict action chunks $\hat{\mathbf{A}}_t^e$ and embodiment latent-token sequences $\hat{\mathbf{Y}}_{t+k}^e$
- 7: Compute dreaming target-token sequences $\mathbf{Y}_{t+k}^e$ with Eq. (6)
- 8: Compute $\mathcal{L}^e$ with Eq. (11) and accumulate its gradient
- 9: **end for**
- 10: Update students with the accumulated gradient of $\mathcal{L}_{\text{pre}}$ in Eq. (12)
- 11: Update teachers with Eq. (9)
- 12: **end for**
- 13: Initialize MXT+ from the pretrained checkpoint
- 14: **for** each fine-tuning step **do**
- 15: Sample a batch from $\mathcal{D}_{e^*}^{\text{task}}$
- 16: Compute $\mathcal{L}_{\text{ft}}$ with Eq. (13) and update students by gradient descent
- 17: Update teachers with Eq. (9) if $\lambda_{\text{dream}}^{e^*} > 0$
- 18: **end for**
- 19: **return** π^{e^*}

Training samples. A training sample is a time step t drawn uniformly from the time steps of the training trajectories. It contains the observations $\mathbf{o}_t^e$, the action chunk of the next $H = 60$ steps, i.e., 2 s at 30 Hz, and the observations of the dreaming modalities $\mathcal{M}_{\text{dream}}^e$ at the offsets $\mathcal{K} = \{20, 40, 60\}$. Chunk steps and dreaming offsets beyond the end of the trajectory are marked as

padding, and action dimensions that an embodiment does not provide are marked by its action mask; both are excluded from the losses. Non-image observations, actions, and dreaming sources are standardized per embodiment and per field with the mean and standard deviation of that embodiment’s demonstrations.

Embodiment-balanced batches. In pretraining, every optimizer step draws a mini-batch of B samples from each embodiment in $\mathcal{E}_{\text{pre}}$, so all embodiments contribute equally to every update regardless of their dataset sizes. The mini-batches are processed one embodiment at a time, and their gradients are accumulated before a single optimizer step. Because each $\mathcal{L}^e$ is averaged over its own mini-batch, the update follows the sum in Eq. (12). Fine-tuning and training without pretraining draw mini-batches from the target-task data only.

Losses. The action loss follows Eq. (10) with $\lambda_m^e = 1$ for every modality and is computed on standardized actions. In our implementation, the L1 norm is replaced by the smooth L1 loss with a threshold of 0.05, which is quadratic for errors below the threshold and linear above it, and the loss is averaged over valid chunk steps and action dimensions. The dreaming loss follows Eq. (8) with $\lambda_{\text{dream}}^e = 0.3$ and is averaged over all latent tokens at dreaming offsets that are not padded; $\beta = 0.01$ for the Kitchen-Bread and Kitchen-Cleanup experiments and $\beta = 0.005$ for the Scoop, Insert, and Sort experiments. For the Kitchen-Bread and Kitchen-Cleanup experiments, $\mathcal{L}^e$ also includes a penalty $10^{-5} \max(0, \|\mathbf{z}_j\|_2 - 100)^2$ on each latent token $\mathbf{z}_j$ produced by the student embodiment tokenizer ϕ_{dream}^e , averaged over valid tokens to discourage large token norms. The teachers are exempt from this penalty.

Optimization. We use AdamW with weight decay 10^{-4} and four parameter groups with separate learning rates: the tokenizers, which include the modality tokenizers, the embodiment tokenizers, and any shared image encoder or tokenizer; the trunk; the action experts; and the dreaming experts. All remaining parameters use the trunk learning rate. The tokenizers, trunk, and action experts share one learning rate, 10^{-4} in pretraining and in training without pretraining and 5×10^{-5} in fine-tuning, while the dreaming experts use a much smaller one (Table 1). Training uses bfloat16 mixed precision without gradient clipping. Pretraining and training without pretraining use a cosine schedule without warmup: at step t of a budget of T steps, every parameter group with initial learning rate η_0 uses

$$\eta_t = \eta_0 \left(r + (1 - r) \frac{1}{2} (1 + \cos(\pi t / T)) \right), \quad r = 0.05, \quad (18)$$

so each learning rate, including that of the dreaming experts, decays to 5% of its initial value at the end of training. Fine-tuning uses constant learning rates.

Image tokenizer sharing. MXT+ shares the image encoder across embodiments, while each embodiment keeps its own adapter, i.e., the cross-attention aggregator g_m^e with its projection and learnable query tokens (Appendix A.2). Encoders are shared per image modality; for example, one encoder processes the egocentric main-camera images of all embodiments. The shared encoder is trained on the images of every embodiment, which pools visual data that is highly imbalanced across embodiments, with human demonstrations making up 49.4% of all trajectories (Table 6), and lets object features learned from one embodiment’s images, such as those of human demonstrations, be reused by the others. The adapters absorb what differs between embodiments: camera placement, field of view, image resolution, and how each embodiment’s own body appears in the image.

EMA teachers. For each embodiment, the teacher consists of copies of the modality tokenizers of $\mathcal{M}_{\text{dream}}^e$ and of the embodiment tokenizer ϕ_{dream}^e . When the image encoder or the image tokenizer is shared across embodiments, its teacher copy is shared as well, while each embodiment keeps a teacher copy of any image module it does not share, such as its own aggregator. Teachers receive no gradients. After every optimizer step, their parameters are updated with Eq. (9) using $\alpha = 0.999$, and their batch-normalization statistics are copied from the students.

Pretraining. Each cross-embodiment pretraining run excludes the target embodiment’s demonstrations of the target task, which are used only for fine-tuning. The pretraining-mixture ablations in Sec. 4 follow the same procedure with fewer embodiments in $\mathcal{E}_{\text{pre}}$: human-only pretraining uses only human demonstrations, and human-plus-robot pretraining adds demonstrations of the other robots but no data from the target embodiment.

Fine-tuning. Fine-tuning loads the weights of a pretraining checkpoint, starts from a new optimizer state, and recomputes the normalization statistics from the target-task data. If the target embodiment is in $\mathcal{E}_{\text{pre}}$, its tokenizers, action experts, embodiment tokenizer, dreaming expert, and teachers are restored together with the trunk and any shared image encoder or tokenizer. Otherwise, as in the human-only and human-plus-robot ablations, only the trunk and the shared image encoder or tokenizer, if one is used, are transferred, and the modules of the target embodiment are trained from their initialization. Wrist cameras that are not used during pretraining receive new tokenizers whose backbones are initialized from ImageNet. Embodiment dreaming remains active during fine-tuning with the same λ_{dream}^* and β .

Training without pretraining and ablations. MXT+ without pretraining is trained from scratch on the target-task data with the same objective, including embodiment dreaming, and with a private image tokenizer. The MXT ablation removes the embodiment tokenizer, the dreaming expert, and the teachers in both pretraining and fine-tuning, and is otherwise trained with the same recipe as MXT+.

A.5 BASELINE IMPLEMENTATIONS

ACT is trained from scratch and $\pi_{0.5}$ is fine-tuned from its pretrained checkpoint, both on demonstrations from the target embodiment and task only. Both are deployed with the execution loop and controllers of Appendix A.8, and Table 2 lists their hyperparameters. MXT+ without pretraining and the MXT ablation are described in Appendix A.4.

ACT. We use the decoder-only variant of ACT (Zhao et al., 2023) introduced in HumanPlus (Fu et al., 2024), which removes the conditional VAE and the encoder-decoder structure of the original model. A single stack of self-attention blocks processes one proprioception token, the image tokens, and 60 learned action queries, and a linear head maps the outputs at the query positions to a chunk of 60 actions. Image tokens are the ResNet-18 feature maps of each camera view, projected to the model width by a 1×1 convolution and combined with 2D sinusoidal positional embeddings. The two halves of the side-by-side stereo head-camera image are encoded as two views by a shared backbone; in Kitchen-Bread and Kitchen-Cleanup, the two wrist cameras are added as views with their own backbones. Proprioceptive states and actions are standardized with dataset statistics. The training loss is an L1 loss on the action chunk, excluding steps padded past the end of an episode, plus an auxiliary feature-prediction loss: the mean squared error between the outputs at the image-token positions and the projected backbone features of the frame 60 steps ahead. ACT has no embodiment tokenizer, action experts, or dreaming expert, and it is not pretrained on other embodiments.

$\pi_{0.5}$. We use the OpenPI implementation of $\pi_{0.5}$, initialized from `pi05_base`. LoRA adapts the language backbone and action expert; the visual encoder, action projections, and time-conditioning MLPs are also trainable. Visual inputs comprise the left stereo view and, for Kitchen tasks, both wrist views, resized with aspect-ratio-preserving padding to 224×224 . Embodiment-specific proprioception is quantile-normalized and discretized into tokens alongside the task instruction. Actions contain end-effector poses with fingertip targets for xArm7-Dex4, finger commands for G1-Dex5, or gripper openings for R1Lite-Gripper and LocoMan-Dex2; Z1-Tool predicts end-effector poses only. G1 and LocoMan also predict torso commands, and Kitchen policies include base velocities. Simulated Dex3 hands use one closure fraction per hand. Selected pose targets are encoded relative to the current observation; simulated Kitchen-Cleanup uses SO(3) rotation vectors for relative orientations. Actions are quantile-normalized, and state and action tensors are zero-padded to 32 dimensions. The model predicts 60-action chunks and is trained with a flow-matching velocity MSE loss. Standard fine-tuning follows Table 2, with 500 warmup steps and global batch sizes. At deployment, all $\pi_{0.5}$ policies use real-time chunking (RTC) (Black et al., 2025c), overlapping prediction of the next action chunk with execution of the current chunk to reduce discontinuities. The inference server uses 10 Euler integration steps per chunk. Outputs are unnormalized and converted to embodiment-specific controller commands, executed at 30 Hz.

Table 1: **Training hyperparameters.** Shared settings apply to all experiments; the remaining blocks list the task-specific settings. Pretraining batch sizes count the samples drawn from each embodiment per optimizer step. Values in parentheses are for R1LITE-GRIPPER. Parameter counts exclude the EMA teachers and cover the trunk together with the modules of every embodiment in pretraining and of the target embodiment otherwise. The largest per-embodiment counts are for *Kitchen-Bread* and *Kitchen-Cleanup*, whose target embodiments add two wrist cameras.

	Pretraining	Fine-tuning	No pretraining
<i>Shared settings</i>			
Optimizer	AdamW, weight decay 10^{-4} , no gradient clipping		
Precision	bfloat16 mixed precision		
Learning-rate schedule	Cosine, no warmup	Constant	Cosine, no warmup
Tokenizer learning rate	10^{-4}	5×10^{-5}	10^{-4}
Trunk learning rate	10^{-4}	5×10^{-5}	10^{-4}
Action-expert learning rate	10^{-4}	5×10^{-5}	10^{-4}
Action horizon H	60 steps (2 s)		
Action loss	Smooth L1 with threshold 0.05 on standardized actions, $\lambda_m^e = 1$		
Dreaming offsets $\mathcal{K}$	{20, 40, 60} steps		
Dreaming loss weight	$\lambda_{\text{dream}}^e = 0.3$		
EMA decay α	0.999		
Image backbone	ResNet-18 initialized from ImageNet		
Parameters	120M–185M	50M–90M	50M–90M
<i>Kitchen-Bread and Kitchen-Cleanup</i>			
Batch size	64 per embodiment	128 (72)	128 (72)
Dreaming-expert learning rate	5×10^{-6} in every stage, with the same schedule		
Dreaming magnitude weight	$\beta = 0.01$		
Latent norm penalty	Bound 100, weight 10^{-5} , in every stage		
<i>Insert</i>			
Batch size	48 per embodiment	32	32
Dreaming-expert learning rate	10^{-6} in every stage, with the same schedule		
Dreaming magnitude weight	$\beta = 0.005$		
Latent norm penalty	None		
<i>Sort</i>			
Batch size	48 per embodiment	32	32
Dreaming-expert learning rate	10^{-6} in every stage, with the same schedule		
Dreaming magnitude weight	$\beta = 0.005$		
Latent norm penalty	None		
<i>Scoop</i>			
Batch size	48 per embodiment	32	32
Dreaming-expert learning rate	10^{-6} in every stage, with the same schedule		
Dreaming magnitude weight	$\beta = 0.005$		
Latent norm penalty	None		

Table 2: **Baseline hyperparameters.** For parameters and batch size, the first value is for Sort, Insert, and Scoop, and the second is for Kitchen-Bread and Kitchen-Cleanup, where ACT adds a ResNet-18 for each of the two wrist cameras.

	ACT	$\pi_{0.5}$
Initialization	ResNet-18 from ImageNet; Transformer from scratch	Pretrained $\pi_{0.5}$ (pi05_base)
Architecture	Decoder-only, 12 blocks, width 512, 8 heads, dropout 0.1	PaliGemma + 300M action expert; LoRA ranks: 16 (language), 32 (action expert)
Parameters	49M; 72M	3.4B
Action horizon	60 steps (2 s)	60 steps (2 s)
Training loss	$L1 + 10^{-3} \times$ feature prediction	Flow-matching velocity MSE
Optimizer	AdamW, weight decay 10^{-4}	AdamW, weight decay 10^{-10}
Learning rate	2×10^{-5} (10^{-5} for ResNet-18), constant	$2.5 \times 10^{-5} \rightarrow 2.5 \times 10^{-6}$, cosine decay
Precision	bfloat16 mixed precision	bfloat16/float32 mixed precision
Batch size	32; 128 (72 for R1Lite-Gripper)	32; 128 (72 for R1Lite-Gripper)

A.6 EMBODIMENT AND HARDWARE DETAILS

EgoX is evaluated on five physical robot embodiments spanning substantially different morphologies, mobility configurations, and manipulation interfaces, as shown in Fig. 8. The platforms include a bimanual humanoid with dexterous hands (G1-DEX5), a wheeled bimanual mobile manipulator with parallel-jaw grippers (R1LITE-GRIPPER), a tabletop manipulator with a dexterous hand (XARM7-DEX4), a fixed-base arm equipped with a non-actuated tool (Z1-TOOL), and a quadruped with a two-finger gripper (LOCOMAN-DEX2).

Across all embodiments, EgoX represents manipulation actions using a shared set of semantically corresponding geometric action modalities. These include end-effector pose targets and, when available, task-relevant end-effector keypoint targets. For dexterous hands and grippers, the keypoints correspond to fingertips, while embodiments that do not provide such structure can use only the applicable subset of action modalities. The predicted geometric targets are mapped to each embodiment’s native control space through embodiment-specific retargeting and low-level control interfaces.

G1-Dex5. G1-DEX5 is a Unitree G1 humanoid equipped with two Inspire RH56F1 five-finger dexterous hands. EgoX can predict body pose, left/right end-effector pose targets together with five fingertip targets for each hand and optional velocity commands for the whole robot. These geometric targets are retargeted to the corresponding arm and hand control spaces for execution.



Figure 8: Physical robot embodiments used in EgoX.

Table 3: **Embodiment-specific interfaces used in EgoX.** EgoX represents manipulation actions using semantically corresponding geometric modalities while allowing each embodiment to expose only the modalities applicable to its morphology. EEf denotes end effector.

Embodiment	Side	Robot Joint Dim.	EEF Joint Dim.	Body Action	EEF Pose Action	Fingertip Targets
G1-DEX5	Bimanual	29	6 + 6	Yes	L/R	5 / hand
R1LITE-GRIPPER	Bimanual	15	1 + 1	No	L/R	2 / gripper
XARM7-DEX4	Right	7	16	No	Right	4
LOCOMAN-DEX2	Right	18	1	Yes	Right	2
Z1-TOOL	Right	6	–	No	Right	–

“Robot Joint Dim.” and “EEF Joint Dim.” denote the native proprioceptive dimensions used by each embodiment. EgoX predicts geometric EEf-pose and, when applicable, fingertip targets, which are subsequently mapped to each embodiment’s native control space.

For locomotion and whole-body stabilization, we use the low-level whole-body controller from Humanoid Touch Dream (Niu et al., 2026), which is kept fixed throughout our experiments.

R1Lite-Gripper. R1LITE-GRIPPER is a Galaxea R1 Lite wheeled bimanual mobile manipulator equipped with parallel-jaw grippers. EgoX predicts left/right end-effector pose targets together with two fingertip targets for each gripper and optional velocity commands. These geometric targets are mapped to the native arm and gripper control spaces of the R1 Lite for execution. Mobile-base commands are handled through the embodiment-specific locomotion interface.

xArm7-Dex4. XARM7-DEX4 consists of an xArm7 tabletop manipulator equipped with a four-finger LEAP Hand (Shaw et al., 2023). EgoX predicts an end-effector pose target together with four fingertip targets. These geometric targets are retargeted to the xArm7 arm and LEAP Hand control spaces for execution. Since the platform has a fixed base, it does not use a locomotion action modality.

Z1-Tool. Z1-TOOL is a Unitree Z1 tabletop manipulator equipped with a fixed, non-actuated scoop. For this embodiment, EgoX predicts only the end-effector pose target, which is executed by the underlying arm controller. Since the scoop is rigidly attached to the arm and has no actuated joints, this embodiment does not use a separate end-effector keypoint or actuation modality.

LocoMan-Dex2. LOCOMAN-DEX2 is a quadrupedal manipulation platform with two-finger grippers mounted on its front legs (Lin et al., 2024). We evaluate the platform in a unimanual configuration. EgoX predicts an end-effector pose target together with two fingertip targets for the active gripper. These geometric targets are mapped to the native control space of the active manipulation limb for execution.

A.7 WORLD AND UNIFIED EGOCENTRIC FRAMES

All frames in EgoX are right-handed, with the x -axis pointing forward, the y -axis to the left, and the z -axis upward, as in IsaacGym and IsaacLab. Each embodiment uses two frames, summarized in Table 4. The world frame $\mathcal{F}_{\text{world}}$ is the reference for the torso pose; it is fixed to the environment for human data and to the part of the robot that rests on the ground (feet, chassis, or arm base) for robots. Except for the fixed-base arms, the wrist poses in $\mathcal{F}_{\text{world}}$ are also provided as proprioception. The unified egocentric frame $\mathcal{F}_{\text{ego}}$ (Sec. 3.1) is rigidly attached to the body that carries the main camera. Wrist poses and wrist targets are expressed relative to it,

$${}^{\text{ego}}T_{\text{wrist}} = \left({}^{\text{world}}T_{\text{ego}} \right)^{-1} {}^{\text{world}}T_{\text{wrist}}, \quad (19)$$

where ${}^AT_B \in SE(3)$ is the pose of frame B in frame A , and end-effector-link keypoints are expressed in the local frame of each end effector. Poses are stored as a position and roll, pitch, and yaw angles in Z – Y – X order. A fixed rotation brings each robot’s native wrist frame to the EgoX wrist convention, $R_{\text{wrist}} = R_{\text{native}}M^\top$, where the rows of M give the native axes that become the

Table 4: **World and unified egocentric frames of each embodiment.** All frames use x forward, y left, and z up. “Torso pose” lists the torso-pose components marked valid in the recorded data (rpy: roll, pitch, and yaw). “Wrist axes” lists the native wrist axes that become the EgoX wrist (x, y, z) axes; for the human, the tracked wrist frame defines the convention.

Embodiment	World frame $\mathcal{F}_{\text{world}}$	Egocentric frame $\mathcal{F}_{\text{ego}}$	Torso pose	Wrist axes
HUMAN-DEX5	Head at episode start, gravity-aligned	Tracked head, carrying the stereo camera	x, y, z, rpy	Tracked wrist
G1-DEX5	Stance frame at the feet, from leg kinematics	Torso link, origin shifted by (0.154, 0, 0.406) m	z, rpy	L: $(x, -z, y)$ R: $(x, z, -y)$
G1-SIM-DEX3	Same as G1-DEX5			
R1LITE-GRIPPER	Wheeled chassis	Floating base (top torso link)	z, pitch	L: $(x, -z, y)$ R: $(x, z, -y)$
LOCOMAN-DEX2	Trunk at mode entry, fixed to the stance feet	Trunk	z, rpy	$(-z, x, -y)$
XARM7-DEX4	Arm base	Arm base (static camera)	–	$(x, -y, -z)$
Z1-TOOL	Arm base rotated by -90° about z	Same as world frame (static camera)	–	(x, y, z)

EgoX wrist x -, y -, and z -axes (Table 4). Torso-pose components that an embodiment does not use are masked (Appendix A.1).

Human-Dex5. The headset reports head and wrist poses in its own gravity-aligned tracking frame, whose y -axis points up; a fixed rotation maps them to the convention above. When the operator starts an episode (Appendix A.9), this frame is rotated about z by the operator’s head yaw and shifted to the head position. $\mathcal{F}_{\text{world}}$ therefore has its origin at the head, its x -axis along the horizontal heading of the head, and its z -axis opposite to gravity, all taken at the start of the episode. $\mathcal{F}_{\text{ego}}$ is the tracked head frame, which carries the headset-mounted stereo camera and follows the full head orientation, including pitch and roll. No head-to-torso offset is applied, so the torso modality of human data is the head pose in $\mathcal{F}_{\text{world}}$. The 25 tracked hand keypoints are expressed in the tracked wrist frame, and the fingertips are taken from them.

G1-Dex5. The controller computes the robot state from the joint encoders by forward kinematics, assuming that both feet are flat on the ground. $\mathcal{F}_{\text{world}}$ is the resulting stance frame, recomputed at every control step: its origin is the midpoint of the two ankle-roll links, its z -axis is the mean of the two foot z -axes, and its x -axis is the mean forward direction of the feet, made orthogonal to z . $\mathcal{F}_{\text{ego}}$ is a virtual head frame rigidly attached to the torso link, which carries the head and its camera. It has the orientation of the torso link, and its origin lies at (0.154, 0, 0.406) m in torso-link coordinates, i.e., (0.15, 0, 0.45) m from the pelvis when the waist joints are at zero. The torso observation is the torso-link pose in $\mathcal{F}_{\text{world}}$. The torso target is the operator’s head pose in the human $\mathcal{F}_{\text{world}}$: the whole-body controller tracks its orientation as the torso orientation and its height change Δz since the reset as a torso height of $0.75 \text{ m} + \Delta z$. Each wrist frame lies 0.05 m beyond the wrist-yaw joint along the joint’s x -axis. For arm inverse kinematics, wrist targets in $\mathcal{F}_{\text{ego}}$ are shifted by the same head offset into the pelvis frame.

G1-Sim-Dex3. The simulated G1 with three-finger Dex3 hands follows the same conventions as G1-DEX5.

R1Lite-Gripper. The vendor motion-control stack reports the pose of the floating base, the top torso link that carries the stereo head camera and both arms, in the frame of the wheeled chassis, and the end-effector poses relative to the floating base. $\mathcal{F}_{\text{world}}$ is therefore the chassis frame, and $\mathcal{F}_{\text{ego}}$ is the floating-base frame. Only the height and pitch of the torso pose are used; the policy predicts no torso target (Table 3), and the chassis is driven by velocity commands. The wrist frames are the vendor end-effector frames, remapped with the same side-specific rotations as G1-DEX5.

LocoMan-Dex2. The robot is operated in its single-gripper manipulation mode, in which the on-board state estimator sets $\mathcal{F}_{\text{world}}$ to the trunk frame at the moment the mode is entered. Afterwards,

the trunk pose is the rigid transform that aligns the current stance-foot positions, obtained from leg kinematics, with their positions at that moment, so $\mathcal{F}_{\text{world}}$ stays fixed to the stance feet. The gripper pose follows from the trunk pose and the forward kinematics of the manipulating leg and its gripper. $\mathcal{F}_{\text{ego}}$ is the trunk frame, which needs no rotation remap in the unimanual configuration. The height and orientation of the torso pose are used.

xArm7-Dex4. The arm has a fixed base and a static main camera, so $\mathcal{F}_{\text{world}}$ and $\mathcal{F}_{\text{ego}}$ coincide with the xArm7 base frame, whose x -axis points forward from the base and whose z -axis points up. The torso pose is identically zero and is not used. The wrist frame is the joint-7 frame without offset, remapped as in Table 4 for the default reset configuration.

Z1-Tool. The Z1 also has a fixed base and a static main camera, so $\mathcal{F}_{\text{world}}$ and $\mathcal{F}_{\text{ego}}$ coincide. Both are the Z1 base frame rotated by -90° about z : the unified x -axis is the native $-y$ -axis, the unified y -axis is the native x -axis, and the z -axis is unchanged. Poses reported by the arm are rotated into this frame, and targets are rotated back before inverse kinematics. The wrist frame is the joint-6 frame without offset or remap. The scoop keypoint lies 0.2 m beyond the joint-6 frame along its x -axis; unlike the keypoints of the other embodiments, it is recorded in $\mathcal{F}_{\text{ego}}$ rather than in the end-effector frame.

From operator motion to robot targets. During teleoperation, the operator’s head and wrist poses are first expressed in the human frames above. For G1-DEX5, the wrist poses relative to the head are sent as wrist targets in the robot’s $\mathcal{F}_{\text{ego}}$. For the other physical robots, the operator’s head and wrist displacements and rotations since the reset are applied to the robot’s torso and end-effector poses at the reset, expressed in the robot’s $\mathcal{F}_{\text{world}}$. For Z1-TOOL, these displacements and rotations are scaled by 0.7 and 0.5, respectively.

A.8 ROBOT CONTROL AND DEPLOYMENT

Shared control stack. On the physical robots, teleoperation and policy deployment share the same control stack (Fig. 9). A high-level process, either the teleoperation interface or the policy, produces geometric targets in the unified egocentric frame: end-effector pose targets, fingertip targets, and, when applicable, body pose and velocity commands. Fixed, embodiment-specific transforms map these targets to the robot’s base frame and end-effector convention, and the targets are sent over CycloneDDS to an embodiment-specific controller that converts them into joint commands. The controller publishes the robot state back, which is recorded during teleoperation and used as proprioceptive input during deployment. Because demonstrations are collected through the same controllers that later execute the policy, recorded actions and policy outputs are interpreted in the same way. Table 5 summarizes the controller of each embodiment.

Policy execution. At deployment, all methods send actions at 30 Hz, matching the recording rate, and use the same embodiment-specific controllers. MXT+ and ACT predict chunks of 60 actions and execute them open loop without temporal ensembling. By default, these policies execute 60 actions (2 s) before the next query; they execute 45 actions per query in simulation (Appendix A.11), as does pretrained MXT+ on R1Lite-Gripper Kitchen-Bread. The $\pi_{0.5}$ policies instead use RTC, overlapping prediction of the next chunk with execution of the current chunk (Appendix A.5). An operator can pause the policy or reset the robot from a keyboard at any time.

Arm control. For G1-DEX5, xARM7-DEX4, and Z1-TOOL, end-effector pose targets are converted into arm joint targets by inverse kinematics (IK), solved at every control step with Pinocchio (Carpentier et al., 2019) and CasADi (Andersson et al., 2019) using IPOPT (Wächter & Biegler, 2006) with at most 50 iterations. The objective is a weighted sum of the translation error, the rotation error, a joint regularizer, and a smoothness term toward the previous solution, with weights (50, 1, 0.02, 0.1) for G1-DEX5 and (30, 1, 0.02, 0.2) for xARM7-DEX4 and Z1-TOOL. For G1-DEX5, only the 14 arm joints are optimized and all other joints are locked. Each solution is then smoothed by a weighted moving average over the four most recent solutions, with weights (0.4, 0.3, 0.2, 0.1) from newest to oldest. R1LITE-GRIPPER uses RelaxedIK (Rakita et al., 2018) for its two 6-DoF arms, and LOCOMAN-DEX2 reuses the whole-body controller and finite-state

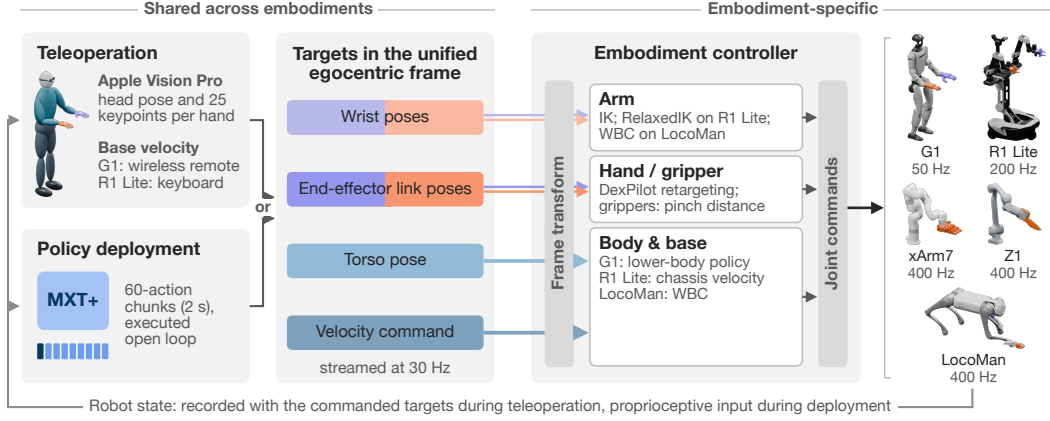


Figure 9: **Shared teleoperation and deployment stack.** Teleoperation and policy rollout produce the same geometric targets, which embodiment-specific controllers convert into joint commands. Target colors follow the action experts in Fig. 2 (purple: left, orange: right), and the rate under each robot is its controller frequency. WBC: whole-body controller.

machine of Human2LocoMan (Niu et al., 2025), which convert the end-effector target of the active manipulator into joint commands.

Hand and gripper control. For the Inspire hands of G1-DEX5 and the LEAP Hand of xARM7-DEX4, fingertip targets expressed in the wrist frame are converted into hand joint commands by DexPilot retargeting (Handa et al., 2020), using an implementation adapted from dex-retargeting (Qin et al., 2023). At every control step, the hand joint angles $\mathbf{q}$ are obtained by solving

$$\min_{\mathbf{q}_{\min} \leq \mathbf{q} \leq \mathbf{q}_{\max}} \frac{1}{N} \sum_{i=1}^N w_i \ell_{\text{smooth-L1}}(\|\mathbf{v}_i(\mathbf{q}) - \hat{\mathbf{v}}_i\|_2) + \lambda_s \|\mathbf{q} - \mathbf{q}_{\text{prev}}\|_2^2 + \gamma \|\mathbf{q}\|_2^2, \quad (20)$$

where $\mathbf{v}_i(\mathbf{q})$ ranges over the N wrist-to-fingertip and fingertip-to-fingertip vectors of the robot hand computed by forward kinematics, $\hat{\mathbf{v}}_i$ is the corresponding target vector, $\ell_{\text{smooth-L1}}$ is the smooth L1 loss with a threshold of 3 cm, and $\mathbf{q}_{\text{prev}}$ is the previous solution. Target vectors are the human vectors scaled by 1.2 for the Inspire hands and 1.6 for the LEAP Hand. As in DexPilot, a thumb-finger pair closer than 3 cm is treated as a pinch until the distance exceeds 5 cm. During a pinch, its target vector is shortened to 0.1 mm so that the fingertips make contact, and two fingers that pinch the thumb together while within 3 cm of each other are kept 3 cm apart. These projected vectors are weighted by $w_i = 200$ and $w_i = 400$, respectively, the remaining fingertip-to-fingertip vectors by $w_i = 1$, and the wrist-to-fingertip vectors by $w_i = 15$ for the Inspire hands and $w_i = 10$ for the LEAP Hand. The second term, with $\lambda_s = 4 \times 10^{-3}$, keeps consecutive solutions close. The last term is not part of dex-retargeting; following DexPilot, we add it to regularize the joint angles toward zero, which corresponds to an open hand. This reduces redundancy in the solution and keeps the hand out of poor local minima that are difficult to recover from, such as fingers embedded in the palm. We use $\gamma = 1.5 \times 10^{-2}$, whereas DexPilot uses $\gamma = 2.5 \times 10^{-3}$ for the Allegro hand. The problem is solved with SLSQP, warm-started from the previous solution, and the resulting joint angles are smoothed by a first-order low-pass filter ($\alpha = 0.2$) before being sent to the hand.

For the grippers of R1LITE-GRIPPER and LOCOMAN-DEX2, the controller computes the distance between the two fingertip targets and maps it linearly to a gripper command, using the same map as teleoperation. On LOCOMAN-DEX2, distances from 2 to 15 cm map to gripper angles from 0.01 to 1.8 rad. On R1LITE-GRIPPER, distances from 0.8 to 15 cm map to the full gripper range; the command is low-pass filtered and then binarized, so the gripper opens fully when the distance exceeds about 5 cm and closes otherwise.

Body and base control. On G1-DEX5, the predicted body pose, consisting of torso height and orientation, and the velocity command, consisting of forward, lateral, and yaw velocities, are passed to the lower-body controller of Humanoid Touch Dream (Niu et al., 2026). This learned policy runs

Table 5: **Low-level control used for teleoperation and deployment.** Every embodiment receives targets at 30 Hz; the listed rate is the frequency of the embodiment controller.

Embodiment	Arm	Hand / gripper	Body / base	Rate
G1-DEX5	IK, 14 arm joints	DexPilot retargeting	Lower-body policy	50 Hz
R1LITE-GRIPPER	RelaxedIK, 2×6 joints	Fingertip distance	Chassis velocity	200 Hz
XARM7-DEX4	IK, 7 joints	DexPilot retargeting	–	400 Hz
LOCOMAN-DEX2	Whole-body controller	Fingertip distance	Whole-body controller	400 Hz
Z1-TOOL	IK, 6 joints	–	–	400 Hz

at 50Hz and outputs targets for the 12 leg joints and 3 waist joints. On R1LITE-GRIPPER, the predicted velocity command drives the mobile chassis and is set to zero whenever the most recent command is older than 0.4 s.

A.9 DATASET AND DATA COLLECTION

Dataset overview. EgoX is trained on a heterogeneous collection of egocentric manipulation demonstrations spanning human, physical-robot, and simulated embodiments. In total, the dataset contains 4,851 trajectories corresponding to 34.38 hours of interaction. Table 6 summarizes the dataset composition for each embodiment, while Fig. 10 visualizes the task distribution across embodiments.

Table 6: **Dataset statistics.** Number of training and validation trajectories and total interaction duration for each embodiment used in EgoX.

Embodiment	Domain	Tasks	Train	Val.	Total	Hours
HUMAN-DEX5	Human	Insert, Kitchen-Bread, Scoop, Sort	2,198	198	2,396	9.26
G1-DEX5	Real	Insert, Kitchen-Bread, Scoop, Sort	622	88	710	6.70
XARM7-DEX4	Real	Insert, Scoop, Sort	398	39	437	3.64
G1-SIM-DEX3	Sim	Kitchen-Bread, Kitchen-Cleanup	691	58	749	7.51
LOCOMAN-DEX2	Real	Scoop	34	3	37	0.35
R1LITE-GRIPPER	Real	Insert, Kitchen-Bread, Sort	398	40	438	6.36
Z1-TOOL	Real	Scoop	77	7	84	0.56
Total	–	–	4,418	433	4,851	34.38

Data composition. The dataset is deliberately heterogeneous in both embodiment and task composition. Human demonstrations constitute the largest source, with 2,396 trajectories (49.4% of the full dataset), while demonstrations collected on physical robots contribute 1,706 trajectories (35.2%). The remaining 749 trajectories (15.4%) are collected with the simulated G1 embodiment. Among the robot embodiments, G1-DEX5 and G1-SIM-DEX3 provide the largest individual datasets, with 710 and 749 trajectories, respectively, while the remaining platforms provide smaller but morphologically diverse sources of manipulation experience.

Importantly, the dataset is not constructed by replicating the same task set across every embodiment. Instead, different embodiments contribute different subsets of tasks according to their morphology and manipulation capabilities. Cross-embodiment overlap therefore arises through semantically corresponding observation and action modalities rather than requiring identical tasks or actuator-level action spaces across all platforms.

Data collection. Similar to Human2LocoMan (Niu et al., 2025), both human and real-robot data are collected with an Apple Vision Pro that captures the operator’s head and hand motions, while simulated data are collected by teleoperation in simulation with PICO XR devices (Appendix A.11). The headset tracks the head pose and 25 keypoints per hand, and all streams are recorded at 30 Hz. Operators start and stop each episode with left-hand pinch gestures, thumb to index finger to start

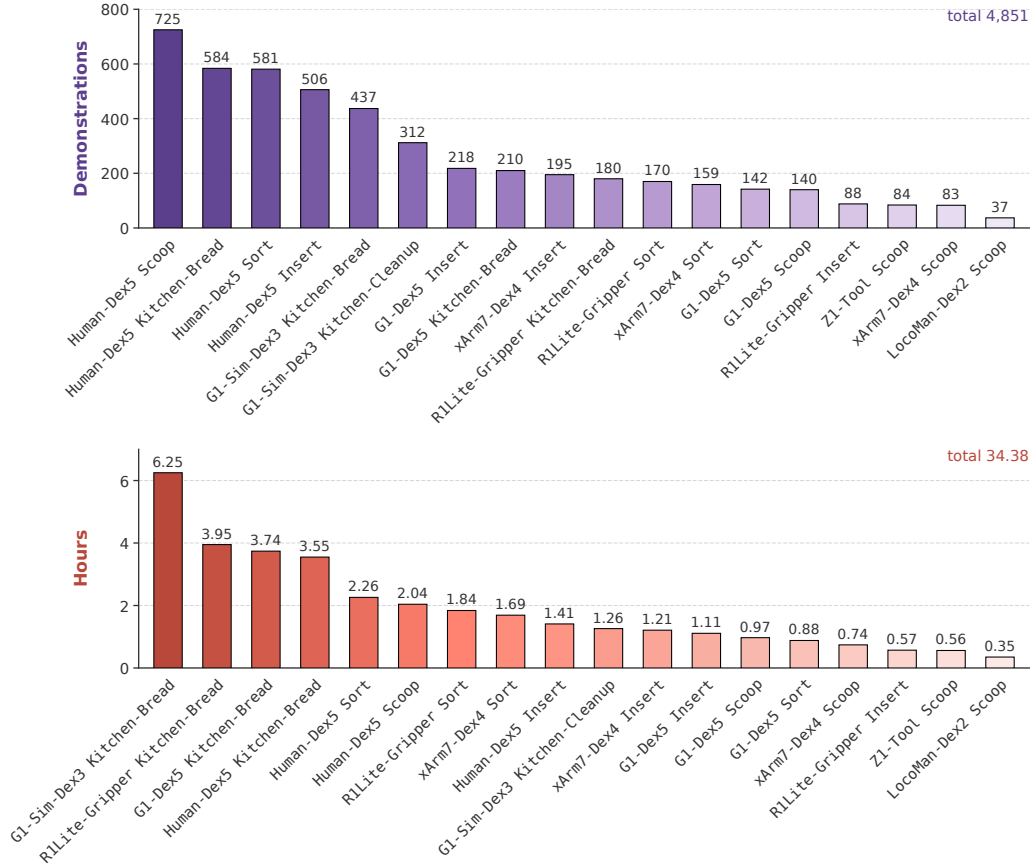


Figure 10: **Composition of the EgoX dataset across embodiments and tasks.** Number of demonstrations (top) and hours (bottom) for each task–embodiment pair, sorted from high to low in each panel.

and thumb to middle finger to stop and reset, so no additional input device is needed. Each episode is stored as an HDF5 file with the images, proprioception, and actions available for that embodiment.

Human demonstrations. For human data, a stereo camera is rigidly mounted on the headset and records egocentric images as side-by-side stereo frames of 1280×480 pixels. The operator performs the task with their own hands while seeing the scene through the headset’s see-through mode (Park & Agrawal, 2024). The tracked wrist poses are expressed in the unified egocentric frame (Sec. 3.1), and the five fingertips of each hand, expressed in the wrist frame, serve as the end-effector link keypoints.

Robot demonstrations. Physical robots are teleoperated with the same headset. The operator’s wrist poses and hand keypoints are converted into targets in the unified egocentric frame and streamed at 30 Hz to the robot, where the control stack of Appendix A.8 executes them. For grippers, the thumb–index pinch distance sets the gripper opening, and for G1-DEX5, the operator’s head orientation and changes in head height set the torso orientation and height. Base velocity commands are issued with a separate input device, the wireless remote of G1-DEX5 or a keyboard for the RLLITE-GRIPPER chassis, and are recorded from the robot state. The operator watches the robot’s stereo head-camera stream in the headset through an Open-TeleVision (Cheng et al., 2024) interface. Each recorded step stores the commanded targets as actions, together with the robot’s egocentric images and the proprioception returned by the controller.

A.10 TASK AND EVALUATION PROTOCOL

We evaluate 13 task–embodiment pairs: *Scoop* on G1-DEX5, xARM7-DEX4, LOCOMAN-DEX2, and Z1-TOOL; *Sort* on G1-DEX5, xARM7-DEX4, and R1LITE-GRIPPER; *Insert* on G1-DEX5 and xARM7-DEX4; *Kitchen-Bread* on G1-DEX5, R1LITE-GRIPPER, and G1-SIM-DEX3; and *Kitchen-Cleanup* on G1-SIM-DEX3. Figs. 11–15 show example rollouts of these pairs, grouped by task.

ID and OOD conditions. Both conditions are defined for each task–embodiment pair with respect to its fine-tuning dataset, i.e., the demonstrations of the target task on the target embodiment, which are excluded from pretraining (Appendix A.4). *In-distribution (ID)* trials use configurations that are seen in this fine-tuning dataset: the same objects, receptacles, and range of initial layouts. *Out-of-distribution (OOD)* trials use configurations that are unseen in the fine-tuning dataset, such as new object shapes, sizes, and colors, a different receptacle, sauce, or plate, or new object positions, while the task goal and the stage scoring remain unchanged. The ID and OOD configurations of each pair are given after the description of each task below.

OOD evaluation tests cross-embodiment transfer. OOD configurations are unseen only by the target embodiment on the target task. They may still appear elsewhere in the pretraining mixture: in the demonstrations of other embodiments, most notably the human demonstrations, or in demonstrations of other tasks. A policy trained only on the target-embodiment data, such as ACT or MXT+ without pretraining, has therefore never observed these configurations, whereas a pretrained policy may have observed them while they were handled by a human or another robot. The OOD evaluation is designed to measure exactly this case: *whether experience with a configuration on other embodiments, especially humans, transfers to a target robot that has never encountered it*. This is the setting that cross-embodiment pretraining targets, and OOD results should be read as a test of cross-embodiment transfer rather than of generalization to configurations that are absent from all training data.

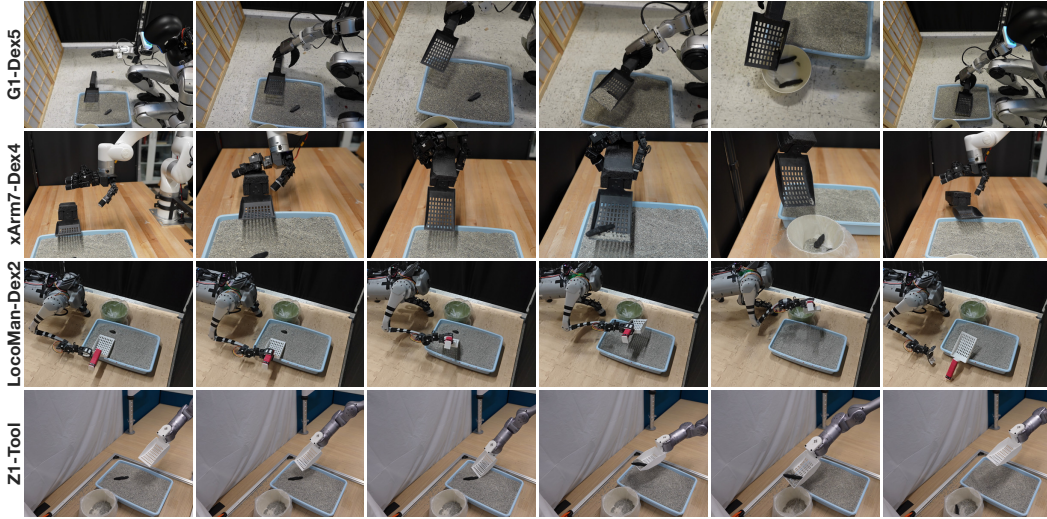


Figure 11: **Scoop across four embodiments.** Each row progresses from left to right through grasping or positioning the scoop, collecting waste, emptying it into the disposal bin, and returning the scoop. Z1-TOOL uses a fixed scoop and does not perform a grasp. The LOCOMAN-DEX2 example shows an OOD trial.

Scoop (cat litter scooping). The scene contains a litter box filled with cat litter, a scoop resting on the edge of the box, and a disposal bin next to it. The robot grasps the scoop, moves it to the waste in the box, i.e., the target material, scoops and lifts it, empties it into the disposal bin, and returns the scoop to the edge of the box. On xARM7-DEX4 and Z1-TOOL, the box stands on a table. On G1-DEX5, it is placed on the floor, so the humanoid lowers its torso to reach it, and LOCOMAN-DEX2 holds the scoop with the gripper on its front leg. Z1-TOOL carries a scoop that is rigidly attached to the arm and therefore skips the grasp. *ID and OOD.* On all four embodiments, ID trials

use a white disposal bin at the position seen in the fine-tuning data, and OOD trials replace it with a green disposal bin or move it to new positions. On G1-DEX5, xARM7-DEX4, and Z1-TOOL, whose workspaces are large, OOD trials also place the waste at new positions in the litter box.

Sort (block sorting). Red and blue building blocks lie on the table in front of a red and a blue bin, whose left-right order varies across trials. The robot picks up each block and places it into the bin of the same color. Each block is scored independently, and approaching or placing into the wrong bin earns only partial credit (Table 7). The bimanual G1-DEX5 and R1LITE-GRIPPER use both arms. *ID and OOD.* ID trials use three blocks of the shapes seen in the fine-tuning data, and each reproduces the initial layout of a recorded demonstration. On all three embodiments, OOD trials use blocks of new shapes, e.g., a four-stud brick. On R1LITE-GRIPPER, OOD trials also change the number of blocks, e.g., to four.

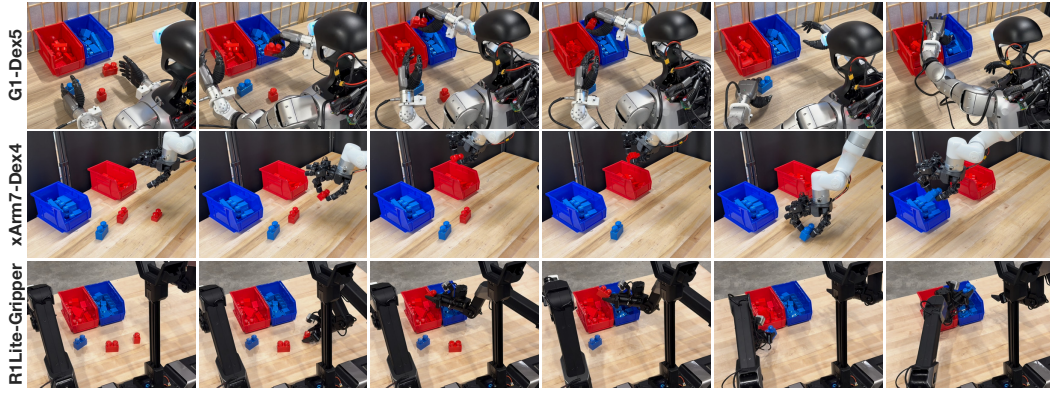


Figure 12: **Sort across three embodiments.** Each row shows the robot grasping blocks and placing them into the color-matched bins, with time progressing from left to right.

Insert (T-shaped block insertion). A T-shaped block lies on the table next to a fixed base with a T-shaped slot. The robot approaches and grasps the block, reorients and aligns it with the slot, and inserts it; a near-successful insertion earns partial credit. *ID and OOD.* ID trials use the block seen in the fine-tuning data, which is purple on G1-DEX5 and white on xARM7-DEX4. On xARM7-DEX4, the block is placed at nine predefined positions, one of which is repeated four more times. OOD trials on G1-DEX5 use a white block or a larger brown block, and OOD trials on xARM7-DEX4 use a purple block or the larger brown block. Each embodiment’s OOD blocks therefore include the block of the other embodiment’s demonstrations, which are part of the cross-embodiment pretraining mixture.

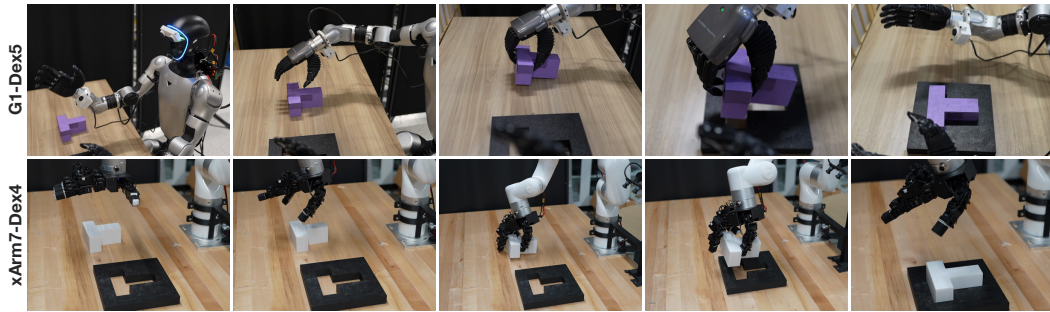


Figure 13: **Insert across two embodiments.** Each row shows the robot approaching, grasping, aligning, and inserting the T-shaped block, with time progressing from left to right.

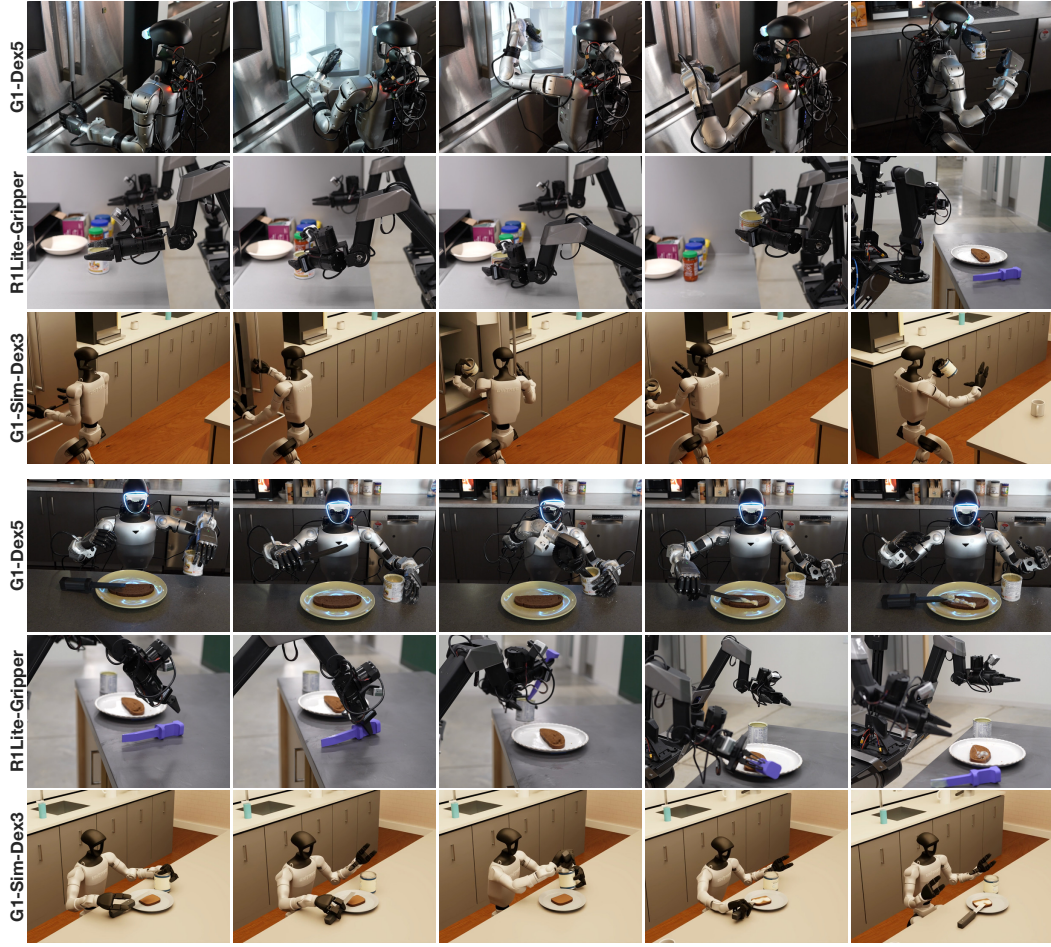


Figure 14: **Kitchen-Bread across three embodiments.** The upper three rows show container pickup and transport; the lower three rows continue the same rollouts through container placement, tool pickup, dipping, spreading, and tool return. Time progresses from left to right within each row. R1LITE-GRIPPER retrieves the container from a counter rather than a refrigerator.

Kitchen-Bread (sauce spreading). This long-horizon task combines locomotion with manipulation. On G1-DEX5, the robot opens the refrigerator, grasps the sauce container, closes the refrigerator, walks to the counter, and places the container next to a plate of bread. It then picks up the spreading tool, dips it into the sauce, spreads the sauce onto the bread, and puts the tool back. On R1LITE-GRIPPER, the robot picks up the sauce container from a counter instead of a refrigerator, drives its wheeled base to the table with the bread, places the container, and performs the same spreading steps. G1-SIM-DEX3 performs the G1-DEX5 sequence in the simulated kitchen (Appendix A.11). *ID and OOD.* On G1-DEX5, ID trials use white condensed milk and brown bread on a green plate, and OOD trials use another kind of condensed milk and a blue plate, with white or brown bread. On R1LITE-GRIPPER, ID trials use white condensed milk and brown bread, and OOD trials use a different sauce, e.g., mayonnaise or strawberry jam, or white condensed milk on white bread; the plate is the same in both conditions. On G1-SIM-DEX3, ID trials use the appearances and layouts randomized in the demonstrations, and OOD trials use real-world-inspired or cross-task appearances and unseen plate-spreader layouts (Appendix A.11).

Kitchen-Cleanup (tool organization). This task is evaluated only on G1-SIM-DEX3. A spreader rests on a plate on the counter next to a container, and the robot grasps the spreader and places it into the container. *ID and OOD.* ID trials use the appearances and layouts randomized in the demonstrations, and OOD trials use real-world-inspired or cross-task appearances (Appendix A.11).



Figure 15: **Kitchen-Cleanup on G1-Sim-Dex3.** From left to right, the robot approaches the spreader, grasps and lifts it, and places it into the container.

Trials and scoring. Each real-world pair is evaluated over 20 trials (13 ID and 7 OOD), and each simulated pair over 50 trials (32 ID and 18 OOD). All methods are evaluated on the same trial configurations. We score each trial with a stage-based task progress score that gives partial credit for intermediate stages. For a trial with accumulated stage score s , success indicator $b \in \{0, 1\}$, and task-specific maximum score $S_{\max}$ including the success bonus, the normalized task progress score is $P = (s + b)/S_{\max}$. Table 7 lists the stages, points, and success criteria of each task. ID and OOD scores are reported separately.

A.11 SIMULATION DETAILS

Simulation setup. We simulate G1-Sim-Dex3, a Unitree G1 humanoid with three-finger Dex3 hands, in a kitchen environment built in Isaac Sim and Isaac Lab. In *Kitchen-Bread*, the robot retrieves a sauce can from the refrigerator, carries it to the worktop, and spreads its contents onto bread; the shorter *Kitchen-Cleanup* task requires picking up the spreader and placing it in a container. Objects interact through rigid-body contact, while sauce transfer is rendered visually without fluid simulation. Demonstrations are collected by teleoperation with PICO controllers, and the egocentric head-camera and wrist-camera images are rendered offline with RTX. Physics runs at 200 Hz and the locomotion controller at 50 Hz.

Randomization. At reset, we randomize the robot and object positions, object sizes, aisle width, and worktop height, keep the spreader supported by the plate, and check each sampled layout for clearance. We also vary the bread shape and the positions of background objects and furniture. During offline rendering, we randomize object and room materials, lighting, camera poses and fields of view, and image properties such as color, noise, blur, and compression. These visual changes leave the recorded motions and action labels unchanged. Each rendered episode uses a fixed visual configuration, with noise varying across frames, and the dataset also retains variants close to the original appearance.

Evaluation. Each task is evaluated on 50 fixed trials: 32 ID and 18 OOD. OOD trials use real-world-inspired or cross-task appearances, and *Kitchen-Bread* OOD trials also include unseen plate-spreader layouts. Trial cases were refined through pilot evaluations and then fixed for all methods. *Kitchen-Bread* and *Kitchen-Cleanup* trials last 90 and 45 simulated seconds, respectively. Actions are executed at 30 Hz. MXT+ and ACT execute 45 actions before each new prediction, while $\pi_{0.5}$ uses RTC (Appendix A.5). *Kitchen-Bread* success requires spreading sauce onto the bread and, at the end of the trial, a released spreader settled on a support. *Kitchen-Cleanup* success requires grasping the spreader and placing it inside the container, released and supported, at any point during the trial. We score nine prerequisite-dependent stages for *Kitchen-Bread* and four for *Kitchen-Cleanup*, plus one point for success, giving maximum scores of 10 and 5; normalizing by these maxima gives the task progress score reported in Sec. 4.

Table 7: **Task progress scoring used for evaluation.** A trial earns the points of every stage it completes and a one-point success bonus if it succeeds. $S_{\max}$ is the full stage score plus the bonus, and the normalized task progress score is $P = (s + b)/S_{\max}$ (Appendix A.10).

Task	Embodiment	Stages and points	$S_{\max}$
Scoop	G1-DEX5, XARM7-DEX4, LOCOMAN-DEX2 Z1-TOOL	Grasp the scoop: 1; move the scoop to the target material: 1; scoop and lift the material: 2; deposit the material into the bin: 1.	5 + 1
		The same stages without the grasp, since the scoop is fixed to the arm: approach: 1; scoop and lift: 2; deposit: 1.	4 + 1
Sort	G1-DEX5, XARM7-DEX4, R1LITE-GRIPPER	Each of the n blocks is scored independently: pick up the block: 1; approach a bin: 3 for the bin of the block’s color and 1 for the other bin; place the block into a bin: 3 for the bin of the block’s color and 1 for the other bin. All trials use $n = 3$ blocks, except R1LITE-GRIPPER OOD trials with more blocks, e.g., $n = 4$; each trial is normalized by its own $S_{\max}$.	$7n + 1$
Insert	G1-DEX5, XARM7-DEX4	Approach the T-shaped block: 1; grasp the block: 2; insert the block into the slot: 2 for a full insertion, 1 for a near-successful insertion, and 0 otherwise.	5 + 1
Kitchen-Bread	R1LITE-GRIPPER	Pick up the sauce container: 1; drive to the table: 1; place the container: 1; pick up the spreading tool: 1; dip the tool into the sauce: 1; spread the sauce onto the bread: 1; put the tool back: 1.	7 + 1
	G1-DEX5, G1-SIM-DEX3	Open the refrigerator: 1; grasp the sauce container: 1; close the refrigerator: 1; walk to the counter: 1; place the container: 1; pick up the spreading tool: 1; dip the tool into the sauce: 1; spread the sauce onto the bread: 1; put the tool back: 1.	9 + 1
Kitchen-Cleanup	G1-SIM-DEX3	Grasp the spreader: 1; lift the spreader: 1; insert the spreader into the container: 1; release the spreader so that it is supported inside the container: 1.	4 + 1

On the physical robots, a trial succeeds, and earns the bonus, exactly when it completes every stage, so $P = 1$ if and only if the trial succeeds. In simulation, Kitchen-Bread succeeds when the sauce added to the bread covers at least 5% of it and the spreader ends released and stably supported, and Kitchen-Cleanup succeeds when the spreader is released and supported inside the container (Appendix A.11). A successful simulated Kitchen-Bread trial therefore need not complete every stage; for example, it may leave the refrigerator open.