WALA: LEARNING EXECUTABLE LATENT ACTIONS FROM ACTION-LABELED DEMONSTRATIONS AND ACTION-FREE VIDEOS

Anonymous authors
Paper under double-blind review

ABSTRACT

Human videos provide rich information about object manipulation at a scale difficult to reproduce with robots, but often lack action annotations that can directly supervise robot policies. Realizing this potential requires distinguishing interaction-relevant changes from static appearance and connecting the learned interaction knowledge to executable robot actions. We present **WALA**, a framework named for *World- and Action-supervised Latent Actions*. Its semantic-geometric latent action model (LAM) learns latent actions by encoding and predicting semantic and geometric changes, emphasizing interactions over static appearance while retaining spatial detail. During robot policy training, the LAM encoder and decoder provide guidance through distillation and visual prediction, respectively. Together with robot action supervision, these signals shape policy-generated latent actions to support world prediction and capture the information needed to directly generate executable robot actions. Because LAM supervision requires no action labels, WALA enables co-training on action-labeled robot demonstrations and task-relevant action-free videos. In simulation experiments, our method achieves **92.33%** average success on RoboTwin and **75.2%** on RoboCasa-GR1-Tabletop, exceeding the previous state of the art on RoboCasa by **5.0** percentage points. WALA also shows strong real-robot performance, further enhanced by task-relevant action-free human videos that improve data efficiency and support target-task zero-shot and few-shot transfer.

1 INTRODUCTION

Generalist robot policies have shown strong performance on diverse manipulation tasks (Black et al., 2025a;b), but collecting the robot demonstrations needed to extend these capabilities across tasks and environments remains costly (Khazatsky et al., 2024). Egocentric human videos can capture manipulation experience without operating a robot (Kareer et al., 2025), with public corpora spanning tens of thousands of hours and beyond (Grauman et al., 2022; 2024; Build AI, 2025a; Hoque et al., 2026). Recorded or estimated human motion can provide action supervision for policy pretraining or co-training with robot demonstrations (Luo et al., 2026a; Lyu et al., 2026). Since many videos contain no action annotations (Ye et al., 2025), the complicated pipeline which includes trajectory estimation, coordinate alignment, or retargeting (Yang et al., 2025; Singh et al., 2025) is required. These costs and cumulative errors raise the question of whether interaction representations can instead be learned directly from unlabeled human videos, without requiring explicit action annotations for robot policy learning.

Latent action models (LAMs) provide a way to learn interaction representations from videos without action labels. By modeling inverse and forward dynamics in observation space, they learn actions in a latent space. These models are commonly constructed directly in the raw pixel space of visual observations (Ye et al., 2025; Routray et al., 2026). However, the redundant background information in raw observations motivates the use of pretrained encoders to construct semantically rich feature spaces, offering a more efficient alternative for latent action learning (Bu et al., 2025b). Nevertheless, semantic information alone is insufficient to accurately capture the spatial changes induced by actions, which are crucial for robot manipulation (Yang et al., 2024a). Meanwhile, these LAMs focus on reconstructing complete future observations in pixel or feature space, rather than specifically

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

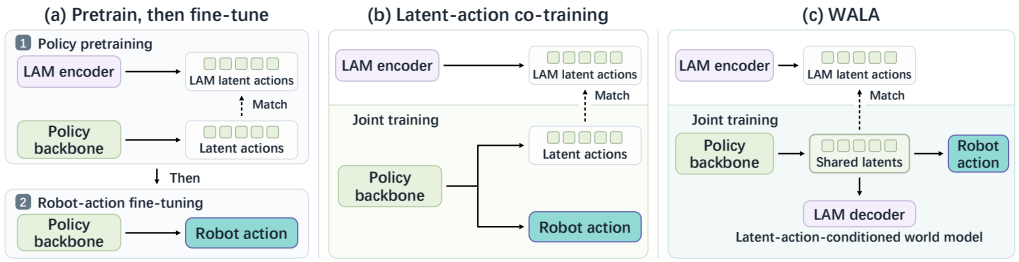

Figure 1: Three paradigms for robot policy learning with a pretrained LAM. (a) A policy is pretrained to predict latent actions and then fine-tuned on action-labeled robot demonstrations. (b) Co-training combines latent-action prediction and robot-action generation within the same policy. (c) WALA combines LAM encoder distillation, LAM decoder world prediction, and robot action supervision through shared policy latent actions that directly feed the action head.

capturing interaction-induced changes. Our goal, however, is for LAMs to represent the actions that drive these changes, rather than the states that result from them.

Once a LAM has learned meaningful interaction representations, the next question is how to effectively transfer this knowledge to robot control. Such transfer can begin by pretraining a policy to predict LAM-inferred latent actions, enabling it to acquire interaction priors from videos without using ground-truth action labels. The pretrained policy is then fine-tuned on action-labeled robot demonstrations (Ye et al., 2025) (Fig. 1(a)). Rather than predicting latent actions only during policy pretraining, co-training combines latent-action prediction with robot action generation within the same policy (Bu et al., 2025b; Chen et al., 2025b), allowing interaction knowledge to continue shaping the policy (Fig. 1(b)). However, using the LAM only to provide encoder-derived latent action targets does not exploit its decoder's potential to guide downstream policy learning through visual prediction. Beyond using the LAM to guide policy learning, we ask whether latent actions themselves can provide the information needed to directly generate executable robot actions.

These considerations highlight two challenges: (1) learning latent actions that capture interaction-relevant changes and spatial geometry, and (2) effectively transferring this knowledge to robot action generation. To address both, we introduce WALA, a framework named for *World- and Action-supervised Latent Actions*. To address the first challenge, WALA's semantic-geometric LAM encodes semantic-feature and depth-map deltas between current and sparsely sampled future observations into latent actions, and uses these latent actions to predict the same deltas. Using temporal differences in both encoding and prediction emphasizes interaction changes over shared static appearance (Wang et al., 2021; Chen et al., 2025a), while pretrained visual features and depth-map supervision provide complementary interaction semantics and fine-grained geometry. To address the second challenge, WALA uses the pretrained LAM together with robot action supervision to shape policy-generated latent actions (Fig. 1(c)). The LAM encoder transfers pretrained interaction knowledge through distillation, while its decoder serves as a latent-action-conditioned world model that further refines these representations through downstream interaction prediction. The action head takes the same latent actions as its sole input, grounding the interaction knowledge they encode in executable robot actions. Since both forms of LAM supervision require no action labels, WALA enables co-training on action-labeled robot demonstrations and task-relevant action-free videos to further improve robot manipulation.

Our contributions are summarized as follows.

- We propose WALA, a framework that enables co-training on action-labeled demonstrations and action-free videos through a shared latent action representation. Its pretrained LAM guides this representation through encoder distillation and decoder prediction, while robot action supervision grounds it in executable robot actions.
- We design a semantic-geometric LAM that learns interaction-relevant latent actions by encoding and predicting semantic-feature and depth-map deltas. This formulation emphasizes interaction changes over shared static appearance, combining semantic abstraction with depth-map supervision to preserve fine-grained geometry.
- We demonstrate strong performance across simulation and real-world manipulation. Our method achieves 92.33% average success on RoboTwin and 75.2% on RoboCasa-GR1-Tabletop. Real-robot experiments further show that task-relevant action-free egocentric human videos improve performance and data efficiency while supporting target-task zero-shot and few-shot transfer.

2 Related Work

2.1 Vision-Language-Action Models and World-Action Models

Vision-language-action (VLA) models such as OpenVLA (Kim et al., 2025), π_0 (Black et al., 2025b), and $\pi_{0.5}$ (Black et al., 2025a) map visual observations and language instructions to actions across tasks and embodiments. QDepth-VLA (Li et al., 2026b) augments VLA training with auxiliary quantized-depth prediction for geometric perception. World-action models (WAMs) jointly predict actions and future observations. LingBot-VA (Li et al., 2026a), DreamZero (Ye et al., 2026b), Motus (Bi et al., 2026), and Fast-WAM (Yuan et al., 2026) predict future RGB observations, typically through compressed video latents, retaining background appearance, texture, and illumination in the prediction targets. LDA-1B (Lyu et al., 2026) instead predicts dynamics in a structured DINO feature space. WALA combines robot action learning with world-model supervision, predicting future DINO-feature and depth-map deltas to capture interaction semantics and local geometry.

2.2 Latent Action Learning from Videos

Latent action models differ in their prediction targets and representations. LAPA (Ye et al., 2025) and Moto (Chen et al., 2025b) learn discrete visual-motion codes through future-frame reconstruction. ViPRA (Routray et al., 2026) augments RGB reconstruction with perceptual and optical-flow consistency losses. UniVLA (Bu et al., 2025b) uses language-conditioned DINO-feature prediction. villa-X (Chen et al., 2026d) couples visual reconstruction with proprioceptive forward dynamics, while JALA (Luo et al., 2026b) aligns policy predictions with inverse-dynamics latents and available actions. UniT (Chen et al., 2026a) combines visual, action, and fused branches through shared quantization and cross-reconstruction. WALA learns latent actions from complementary semantic-feature and depth-map deltas between the current and multiple sparsely sampled future frames.

The learned latent actions are also incorporated into robot policy learning in different ways. LAPA (Ye et al., 2025) directly fine-tunes on robot actions, whereas UniVLA (Bu et al., 2025b) retains latent-action prediction with embodiment-specific action decoding. Moto (Chen et al., 2025b) jointly fine-tunes latent-motion and robot-action prediction. UniT (Chen et al., 2026a) provides latent action token targets for VLA learning and action conditions for world modeling. ViPRA (Routray et al., 2026) jointly pretrains visual-token and latent-action prediction before learning robot actions through flow matching. WALA applies encoder distillation and semantic-geometric decoder prediction to policy-generated latent action tokens that directly feed the action head.

3 Method

3.1 Overview

Our goal is to learn a policy from action-labeled robot demonstrations and task-relevant action-free videos, mapping current observations and instructions to executable actions. WALA connects these data sources through shared latent actions in two stages (Fig. 2). Stage I pretrains a semantic-geometric LAM to encode and predict visual changes without action labels. Stage II co-trains the policy on both data sources, shaping policy-generated latent actions through LAM encoder distillation, LAM decoder prediction, and robot action supervision.

The policy receives multi-view RGB observations, while the LAM uses only the primary camera view. Let I_t denote its current RGB image and D_t its normalized depth map from frozen Depth Anything V2 (Yang et al., 2024b) (Appendix A.4). We sample $K = 4$ future observations $\{o_{t+\tau_k}\}_{k=1}^{K}$, where $o_{t+\tau_k} = (I_{t+\tau_k}, D_{t+\tau_k})$, at evenly spaced offsets within each clip. During robot policy training, human and robot clips cover the same temporal extent as an action chunk (Appendix A.2). A frozen DINOv3 (Siméoni et al., 2026) encoder ϕ extracts patch-level semantic features,

$$X_t = \phi(I_t), \quad X^+_{t,k} = \phi(I_{t+\tau_k}). \tag{1}$$

We write $X^+_t = \{X^+_{t,k}\}_{k=1}^{K}$ and $D^+_t = \{D^+_{t,k}\}_{k=1}^{K}$ for the future features and depth maps, with $D^+_{t,k} = D_{t+\tau_k}$. The corresponding deltas are $\Delta X_{t,k} = X^+_{t,k} - X_t$ and $\Delta D_{t,k} = D^+_{t,k} - D_t$, collected in ΔX^+_t and ΔD^+_t. Throughout, $+$ denotes future observations, whereas a hat denotes a model prediction.

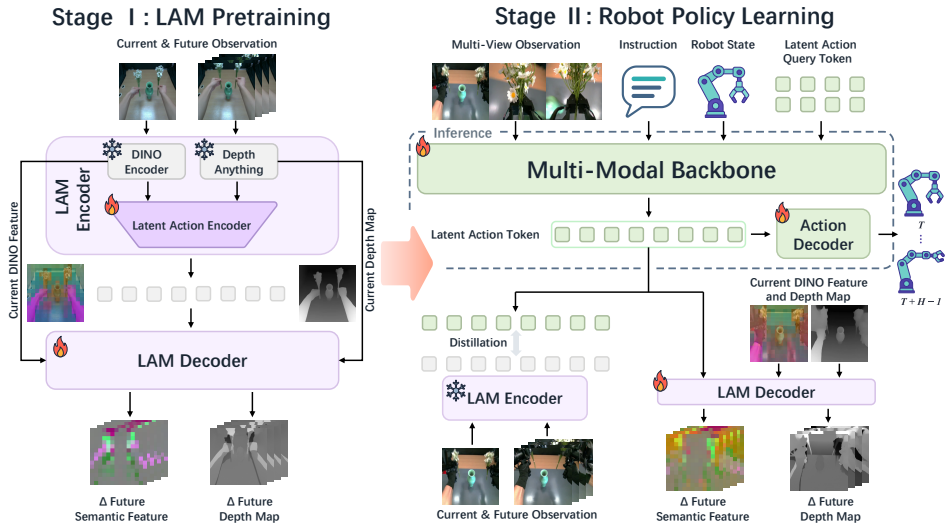

Figure 2: Overview of WALA. Stage I pretrains the semantic-geometric LAM to encode and predict changes between current and sparsely sampled future primary-view observations. Stage II jointly shapes policy-generated latent actions through LAM encoder distillation, LAM decoder prediction, and robot action supervision. These tokens form the action head's sole input. The dashed box marks the inference pathway. Future observations and the LAM are used only during training. Snowflakes and flames denote frozen and trainable modules, respectively.

3.2 Stage I: LAM Pretraining

The semantic-geometric LAM learns interaction-relevant latent actions by encoding and predicting temporal deltas. Its encoder E_θ processes the current observation and future deltas through semantic and geometric branches, then fuses them into latent action tokens,

$$z_t = E_\theta(X_t, D_t, \Delta X_t^+, \Delta D_t^+). \tag{2}$$

The semantic branch operates in patch-level DINOv3 feature space, while the geometric branch operates directly in depth-map space. The representation z_t comprises 32 tokens of dimension 1,024, compressing the changes across all four future observations. These tokens summarize visual changes over the sampled clip; they are not per-step robot commands. Their mapping to executable control is learned during policy training.

The LAM decoder G_ψ predicts future deltas conditioned on both the current observation and z_t,

$$\widehat{\Delta X}_{t,k} = G_\psi^{\text{sem}}(X_t, z_t, k), \quad \widehat{\Delta D}_{t,k} = G_\psi^{\text{dep}}(D_t, z_t, k). \tag{3}$$

Here, θ and ψ parameterize the encoder and decoder; $k \in \{1, \ldots, K\}$ indexes the future observation at $t + \tau_k$. Both branches use latent-action-conditioned feature-wise linear modulation (FiLM) (Perez et al., 2018) to modulate their respective current features. By reusing the current observation, this design reduces the need to encode shared scene content in the latent actions and encourages them to represent changes relative to the current state. Adding the predicted deltas to the current semantic features and depth maps yields their future counterparts. DINOv3 feature prediction provides semantic supervision without future RGB reconstruction, while direct depth-map prediction supplies spatially detailed geometric supervision.

For brevity, ΔX and ΔD below collect all sampled future deltas, with time and future-set indices omitted. The semantic objective combines element-wise regression with cosine loss, which additionally constrains the direction of feature change,

$$\mathcal{L}_{\text{sem}} = \mathcal{L}_{\text{reg}}^{\text{sem}}(\widehat{\Delta X}, \Delta X) + \lambda_{\cos}\mathcal{L}_{\cos}. \tag{4}$$

The depth objective combines regression with depth-gradient consistency,

$$\mathcal{L}_{\text{dep}} = \mathcal{L}_{\text{reg}}^{\text{dep}}(\widehat{\Delta D}, \Delta D) + \lambda_{\text{grad}}\mathcal{L}_{\text{grad}}. \tag{5}$$

The gradient term matches horizontal and vertical finite differences of the predicted and target depth deltas to supervise local depth variations. Losses are averaged over future steps and spatial locations. The coefficients $\lambda_{\cos}$ and λ_{grad} weight these auxiliary terms. The LAM pretraining objective is

$$\mathcal{L}_{\text{LAM}} = \mathcal{L}_{\text{sem}} + \lambda_{\text{dep}}\mathcal{L}_{\text{dep}}. \tag{6}$$

Here, λ_{dep} weights depth relative to semantic prediction. Exact loss definitions and all weights are given in Appendix A.4.

3.3 Stage II: Robot Policy Learning

After pretraining, we freeze the LAM encoder to provide a stable latent action target space. It encodes each sample's observed changes as a distillation target,

$$z_t^{\text{ref}} = \text{sg}\left[E_\theta(X_t, D_t, \Delta X_t^+, \Delta D_t^+)\right], \tag{7}$$

where $\text{sg}[\cdot]$ denotes stop-gradient.

The policy combines a multimodal backbone with an action-query interface. Backbone variants are described in Appendix A.5. Given multi-view RGB observations o_t^{mv}, a language instruction l, robot state s_t when available, and latent action queries q, the policy backbone Π_ω produces H latent action tokens, one per step of an action chunk. The action head A_ω takes only these tokens as input,

$$\widehat{a}_{t:t+H-1} = A_\omega(\widetilde{z}_t), \quad \widetilde{z}_t = \Pi_\omega(o_t^{\text{mv}}, l, s_t, q). \tag{8}$$

Here, ω collects backbone and action-head parameters, and a tilde identifies policy-generated latent actions. Two learned adapters align the token count and feature dimension of $\widetilde{z}_t$ with those of the LAM latent actions for encoder-target distillation and decoder conditioning (Appendix A.3):

$$\bar{z}_t^{\text{enc}} = P_{\text{enc}}(\widetilde{z}_t), \qquad \bar{z}_t^{\text{dec}} = P_{\text{dec}}(\widetilde{z}_t). \tag{9}$$

Writing $U = \bar{z}_t^{\text{enc}}$ and $V = z_t^{\text{ref}}$, the distillation loss is

$$\mathcal{L}_{\text{distill}} = \ell_{\cos}(U, V) + 0.1\,\text{SmoothL1}(\text{LN}(U), \text{LN}(V)), \tag{10}$$

where $\ell_{\cos}$ averages token-wise cosine distance and LN denotes layer normalization over each token's features (Appendix A.4). The pretrained LAM decoder serves as a latent-action-conditioned semantic-geometric world model. It receives $\bar{z}_t^{\text{dec}}$ in place of z_t in Eq. (3) and is jointly fine-tuned with the policy. Its loss $\mathcal{L}_{\text{wm}}$ has the same form and internal weights as $\mathcal{L}_{\text{LAM}}$ in Eq. (6), evaluated on predictions conditioned on policy-generated rather than LAM-encoded latent actions. Distillation transfers pretrained interaction knowledge, while decoder prediction supervises the interaction outcomes implied by policy-generated latent actions. Both losses refine $\widetilde{z}_t$ through the adapters. With the action head reading only these tokens, world and robot action supervision jointly shape the representation used for interaction prediction and executable action generation.

At the sample level, the available supervision is expressed as

$$\mathcal{L}_{\text{policy}} = m\mathcal{L}_{\text{act}} + \lambda_{\text{distill}}\mathcal{L}_{\text{distill}} + \lambda_{\text{wm}}\mathcal{L}_{\text{wm}}. \tag{11}$$

The indicator m is 1 for samples with robot action labels and 0 otherwise. The action loss $\mathcal{L}_{\text{act}}$ is mean L1 error on action chunks. In minibatch optimization, action losses are averaged over labeled samples, whereas both LAM losses are averaged over all samples. The latter support co-training on action-labeled demonstrations and task-relevant action-free videos, jointly refining the latent actions used for robot control. Future observations supervise training but never enter the policy backbone. The coefficients λ_{distill} and λ_{wm} weight the auxiliary losses; their selection and the minibatch reductions are detailed in Appendix A.4.

3.4 Training and Inference

Stage I optimizes the LAM encoder, fusion module, and decoder while keeping DINOv3 and the depth estimator frozen. Stage II keeps the LAM encoder frozen and jointly optimizes the decoder, adapters, policy backbone, and action head, adapting the world model to policy tokens and downstream observations. At inference, only the policy backbone and action head are used to map current multi-view RGB observations, language, available robot state, and action queries to robot actions.

4 Experiments

Our experiments address four questions: (1) How does WALA compare with VLA-, WAM-, and LAM-based policies? (2) How does performance scale with action-labeled demonstrations and action-free videos? (3) What interaction and action information do the latent actions retain, and how do their supervision and action readout contribute to robot control? (4) Can WALA achieve effective and efficient real-robot manipulation and use task-relevant action-free videos to improve performance and transfer?

4.1 Experimental Setup

Standard WALA uses Qwen3-VL-4B-Instruct (Bai et al., 2025); WALA-Wan uses Wan2.2-TI2V-5B (Wan-AI, 2025). RoboTwin (Chen et al., 2026c) uses 50 Clean and 500 Random demonstrations per task across 50 tasks, with 100 evaluation episodes per task in each setting. RoboCasa-GR1-Tabletop (Nasiriany et al., 2024) uses 1,000 demonstrations and 50 evaluation episodes for each of 24 tasks. Our action-only base policy shares WALA's backbone and robot demonstrations, isolating the contribution of LAM supervision.

4.2 Main Benchmark Results

Table 1: Main results on RoboTwin. Baseline results are collected from publicly released reports. Avg. is the mean of the Clean and Random success rates.

Group	Method	Clean	Random	Avg.
VLA-based	$\pi_{0.5}$ (Black et al., 2025a)	82.74	76.76	79.75
	X-VLA (Zheng et al., 2026)	72.80	72.84	72.82
	StarVLA-α (Ye et al., 2026a)	88.20	88.30	88.25
	InternVLA-A1 (Cai et al., 2026)	89.40	89.64	89.52
WAM-based	Motus (Bi et al., 2026)	88.66	87.02	87.84
	Fast-WAM (Yuan et al., 2026)	91.88	91.78	91.83
	LingBot-VA (Li et al., 2026a)	**92.93**	91.55	<u>92.24</u>
Ours	WALA	90.62	**92.80**	91.71
	WALA-Wan	<u>92.36</u>	<u>92.30</u>	**92.33**

Table 2: Main results on RoboCasa-GR1-Tabletop. † marks our reproductions; other baselines are from public reports.

Group	Method	Avg.
VLA-based	StarVLA Qwen3-PI (StarVLA Community, 2026)	43.9
	GR00T-N1.6 (NVIDIA et al., 2025)	47.6
	StarVLA Qwen3-GR00T (StarVLA Community, 2026)	47.8
	StarVLA Qwen3-OFT (StarVLA Community, 2026)	48.8
	StarVLA-α (Ye et al., 2026a)	57.3
	ABoT-M0 (Yang et al., 2026)	58.3
	RLDX-1 (Kim et al., 2026)	58.7
	FrameSkip (Yu et al., 2026)	59.5
WAM-based	DiT4DiT (Ma et al., 2026)	50.8
	LDA-1B (Lyu et al., 2026)	55.4
	DIAL (Chen et al., 2026e)	<u>70.2</u>
LAM-based	Moto (Chen et al., 2025b)†	45.6
	UniVLA (Bu et al., 2025b)†	51.2
	VLA-UniT (Chen et al., 2026a)	66.7
Ours	WALA	**75.2**

RoboTwin. WALA-Wan leads all VLA- and WAM-based baselines with 92.33% average success (Table 1); both variants exceed 92% in Random. LAM encoder distillation transfers pretrained interaction knowledge, while LAM decoder prediction refines it using downstream observations. Joint robot action supervision grounds this interaction-informed representation in the actions generated by the policy. Semantic-geometric delta modeling reduces emphasis on shared background appearance, helping maintain performance under changes in backgrounds, lighting, and distractors. The strong results with both backbones support this training framework across vision-language and video policies.

RoboCasa-GR1-Tabletop. WALA reaches 75.2%, leading all three baseline groups (Table 2) and surpassing the previous state-of-the-art method, DIAL, by 5.0 percentage points. Among LAM-based methods, WALA exceeds VLA-UniT by 8.5 points and outperforms our Moto and UniVLA reproductions under the same GR1 data and evaluation protocol. Unlike RoboTwin, where strong baselines already exceed 90%, RoboCasa leaves more room for improvement. These tasks require identifying the instructed object and receptacle, then placing the object correctly. Semantic supervision supports target identification, while depth supervision represents object–receptacle geometry. Rich human hand–object interactions in pretraining also offer transferable manipulation knowledge for GR1's dexterous hands. WALA also exceeds LDA-1B (55.4%), which directly pretrains robot policies on large-scale human videos with aligned action annotations, demonstrating effective use of human experience without human action labels.

4.3 Scaling with Robot Demonstrations and Action-Free Videos

Robot-demonstration scaling. With 10% action-labeled demonstrations, WALA achieves 53.9% success, compared with 18.1% for the base policy at the same budget, nearly matching the base policy's 54.2% with all demonstrations (Fig. 3, left). With limited robot demonstrations, action supervision alone may be insufficient to learn rich interaction representations. Knowledge learned from videos and supervision of semantic and geometric changes provide additional information about object motion and spatial relationships, helping the policy make better use of the available robot demonstrations. As more action-labeled demonstrations become available, both policies improve and the base policy narrows the gap, consistent with robot data providing increasingly broad coverage of these interactions. Nevertheless, WALA retains a 21.0-point advantage at the full budget,

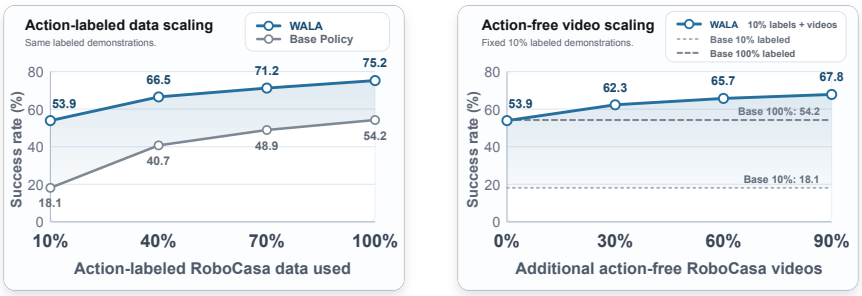

Figure 3: Data scaling on RoboCasa-GR1-Tabletop. Left: WALA consistently improves over the base policy when both use the same number of action-labeled demonstrations. Right: with 10% action-labeled demonstrations fixed, additional videos used without action supervision further improve WALA. All data fractions refer to the full RoboCasa training set.

showing that visual interaction supervision remains beneficial even with substantially more robot action data.

Video scaling without action supervision. With action-labeled demonstrations fixed at 10%, success increases steadily as more videos are added (Fig. 3, right). These videos broaden the object configurations and interaction stages encountered during policy training, extending visual supervision beyond the small action-labeled subset. This richer experience helps the policy learn task-relevant interaction patterns across different manipulation examples, while the robot action supervision budget remains unchanged. The resulting 13.9-point gain demonstrates the value of co-training on action-labeled demonstrations and action-free videos. Both panels show diminishing returns, consistent with increasing overlap among examples of the same tasks, but videos continue to provide useful supervision at every increment.

Table 3: Ablation study on RoboCasa-GR1-Tabletop.

Variant	Supervision				Avg.	Gain
	LAM Pretraining	Latent Distill.	Semantic Feature Pred.	Depth-Map Pred.		
Base Policy	–	–	–	–	54.2	–
+ World Model without Pretraining	–	–	✓	✓	67.8	+13.6
+ Semantic-Feature Prediction	✓	–	✓	–	68.8	+14.6
+ Semantic-Geometric World Model	✓	–	✓	✓	71.0	+16.8
+ Latent-Action Distillation	✓	✓	–	–	67.6	+13.4
WALA	✓	✓	✓	✓	**75.2**	**+21.0**

4.4 Ablation Study

Table 3 uses our action-only base policy (54.2%) with the same policy backbone and robot data. Distillation uses $\mathcal{L}_{\text{distill}}$; semantic and geometric prediction form $\mathcal{L}_{\text{wm}}$ (Sec. 3.3). All rows retain action supervision; variant configurations are detailed in Appendix A.3. World-model supervision reaches 67.8% without LAM pretraining and 71.0% with pretraining, showing that downstream prediction benefits from prior interaction knowledge. The semantic-geometric variant outperforms the variant using a separately pretrained semantic-only LAM by 2.2 points, supporting geometry-aware latent action learning for spatial manipulation. WALA reaches 75.2%, exceeding distillation alone by 7.6 points and world-model prediction alone by 4.2 points. Encoder targets transfer pretrained knowledge, while decoder supervision refines the action-generating tokens through prediction of downstream interactions. Their combined gain shows that encoder-derived targets and downstream visual prediction provide complementary guidance for the same action-generating representation. On RoboDojo videos unseen during LAM pretraining, decoder supervision also reduces semantic and depth prediction errors relative to distillation alone; fine-tuning the decoder improves them further (Appendix B.4). We further examine the action-readout design (Appendix B.5). Reading all final-layer VLM tokens through a learned resampler yields 73.3% success, compared with 75.2% for WALA, providing no additional benefit from the broader feature access. Latent action tokens also receive the highest attention in this variant, supporting their role as a compact multimodal representation for direct robot action generation.

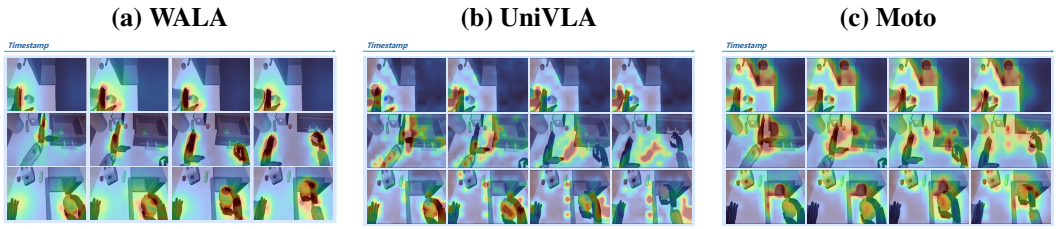

Figure 4: Latent-action encoder attention from WALA, UniVLA, and Moto. Corresponding rows show the same four future frames of one RoboCasa task.

4.5 Analysis of Learned Latent Actions

We first examine the policy-generated latent actions used for robot control. In Appendix Fig. 6, the decoder's semantic and depth predictions capture hand and object movements together with changing spatial structure, showing that the action-generating tokens retain both forms of information. With the current observation fixed, swapping tokens produces incorrect semantic and geometric changes; swapping or zeroing them increases prediction errors (Appendix B.2). Thus, the tokens retain information about future interactions that the decoder cannot obtain from the current scene alone.

We next examine the interaction representations learned by the LAM encoder. WALA's attention concentrates on hands, manipulated objects, and their interaction regions, whereas UniVLA's and Moto's are more dispersed across the scene (Fig. 4). This interaction focus is consistent with delta modeling and helps explain WALA's performance amid visual distractors. Latent-action retrieval recovers similar manipulation primitives across objects, layouts, and backgrounds (Appendix B.1). Action probes of the frozen LAM show that WALA latent actions support more accurate robot action decoding than UniVLA and Moto latent actions and purely visual feature sequences (Appendix B.3). These results indicate that WALA's latent actions retain not only world interaction information but also information that can be effectively mapped to executable robot actions.

4.6 Real-World Experiments

Table 4: Real-world results on 4 base tasks, with 30 trials per task.

Group	Method	Robot demos per task	Human videos per task	Basic Pick-Place	Stack Paper Cups	Insert Flowers	Disassemble Blocks	Avg.	Inference latency (ms)
Baselines	π_0 (Black et al., 2025b)	200	0	25/30	15/30	12/30	10/30	51.7	170
	$\pi_{0.5}$ (Black et al., 2025a)	200	0	26/30	15/30	16/30	13/30	58.3	190
Ours 50 demos	WALA	50	0	26/30	18/30	17/30	15/30	63.3	**70**
	WALA	50	150	28/30	21/30	18/30	17/30	70.0	**70**
	WALA	50	400	**30/30**	23/30	19/30	17/30	74.2	**70**
Ours 200 demos	WALA	200	0	28/30	21/30	21/30	20/30	75.0	**70**
	WALA	200	400	**30/30**	**25/30**	**24/30**	**21/30**	**83.3**	**70**

Experimental setup. We evaluate on an ARX robot. Policies for base-task comparisons are jointly trained on only the 4 base tasks (Table 4), whereas transfer policies are jointly trained using the available data from a 10-task set (Table 5). Additional details on task co-training and evaluation are provided in Appendix A.6.

Performance and inference latency. With 200 demonstrations and no human videos per base task, WALA reaches 75.0% average base-task success, versus 51.7% for π_0 and 58.3% for $\pi_{0.5}$ (Table 4). Even with 50 demonstrations, WALA exceeds both baselines trained on 200. Pretrained interaction knowledge and downstream future-prediction supervision help the policy anticipate object motion and state changes, while depth supervision supports precise spatial alignment. These complementary cues help explain the larger gains over $\pi_{0.5}$ on stacking, insertion, and disassembly than on basic pick-place. All inference latencies are measured per prediction on the same RTX 4090. WALA takes 70 ms using only the policy backbone and action head, achieving roughly a $2.5\times$ inference speedup over π_0 and $\pi_{0.5}$.

Human-video scaling. With 50 demonstrations per base task, adding human videos steadily raises success from 63.3% to 74.2% (Fig. 5(b)), nearly matching 75.0% with 4 times as many robot demonstrations per base task. Human videos also improve performance with 200 demonstrations, reaching 83.3%. Additional human videos from similar scenes broaden the coverage of object positions and

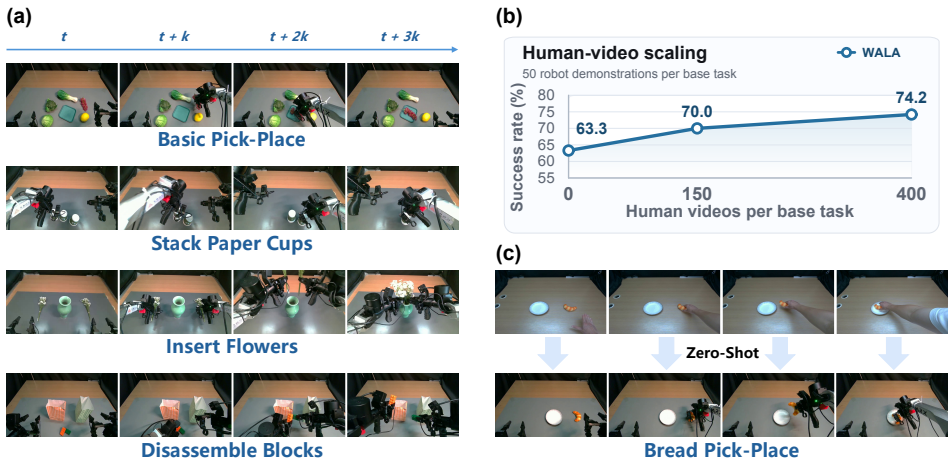

Figure 5: Real-world experiments. (a) Successful rollouts on the 4 base tasks. (b) Human-video scaling with 50 robot demonstrations per base task. (c) Target-task zero-shot transfer to Bread Pick-Place using action-free egocentric human videos and no target-task robot demonstrations.

manipulation patterns, helping the policy handle more varied task configurations without requiring additional robot demonstrations.

Target-task zero-shot transfer. With no target-task robot demonstrations, human videos raise success across all 3 tasks (Table 5). Transfer is strong for single-object pick-place (Fig. 5(c)) and cabinet opening, while repeated 2-object placement remains more challenging. Target-task zero-shot performance may also benefit from action supervision on other co-trained robot tasks. Human videos help the policy understand how new target objects are manipulated, guiding the reuse of related action patterns in new interactions rather than learning new action primitives from video alone.

Table 5: Transfer results as successful trials out of 30. Human-video counts are per target task.

Task	Human videos	
	0	150
Few-shot (5 robot demos/task)		
Turn On Lamp	13/30	**22/30**
Open Shelf-Unit Door	8/30	**19/30**
Pen Pick-Place + Door Closing	6/30	**18/30**
Zero-shot (0 robot demos/task)		
Bread Pick-Place	9/30	**24/30**
Open Wooden Cabinet Door	4/30	**15/30**
Place Two Apples on Plate	0/30	**8/30**

Few-shot transfer. Human videos also improve few-shot performance in switch operation, door opening, and combined pick-place and door closing. The gains extend beyond isolated grasping to object-state changes and sequential manipulation. A few robot demonstrations provide task-specific action guidance, while human videos supplement these limited examples with variations in object placement and motion during task execution.

5 Conclusion and Limitations

WALA connects interaction learning from action-free videos with robot control through a shared latent action representation. Semantic-geometric delta modeling emphasizes interaction changes while retaining spatial detail. Beyond matching pretrained targets, joint world and action supervision refines this representation through downstream prediction and grounds it in executable actions. Latent actions thus connect interaction prediction with action generation. Co-training thereby combines broader interaction experience from action-free videos with the action grounding supplied by robot demonstrations. Experiments show strong performance and data efficiency across simulation and real robots, with task-relevant human videos further improving manipulation and supporting target-task zero-shot and few-shot transfer.

Due to computational constraints, we have not systematically explored scaling the LAM's pretraining data and model size. Large-scale robot-policy pretraining on both action-labeled demonstrations and action-free videos also remains unexplored. Investigating these directions could further improve generalization across tasks, environments, and embodiments.

AI USE STATEMENT

We used generative AI tools for language polishing, literature search, and drafting sections of the manuscript. The authors reviewed all AI-assisted content and take responsibility for the final paper.

ETHICS STATEMENT

This work studies robot manipulation using publicly available datasets, robot demonstrations, and egocentric human videos. The public datasets were used in accordance with their applicable licenses and terms of use. The human videos collected for this study depict task-relevant object manipulation and were recorded with the participants' informed consent. Real-robot experiments were conducted under human supervision.

REPRODUCIBILITY STATEMENT

The model and training objectives are described in Section 3, and the experimental settings and evaluation protocols in Section 4. Appendix Table 6 summarizes the pretraining data composition, and Appendix A.3 provides the model and training configurations. Per-task simulation results are reported in Appendix Tables 9 and 10. The project website, code, and model weights are available anonymously at `https://anonymous.4open.science/w/WALA-Website-CC6F`.

REFERENCES

AgiBot World Team. Agibot world 2026. Hugging Face dataset, 2026. URL `https://huggingface.co/datasets/agibot-world/AgiBotWorld2026`.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025. URL `https://arxiv.org/abs/2511.21631`.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, Hongyan Zhao, Hanyu Liu, Zhizhong Su, Lei Ma, Hang Su, and Jun Zhu. Motus: A unified latent action world model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 35101–35113, June 2026. URL `https://openaccess.thecvf.com/content/CVPR2026/html/Bi_Motus_A_Unified_Latent_Action_World_Model_CVPR_2026_paper.html`.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 2025a. URL `https://proceedings.mlr.press/v305/black25a.html`.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics:*

Science and Systems, Los Angeles, CA, USA, June 2025b. doi: 10.15607/RSS.2025.XXI.010. URL https://www.roboticsproceedings.org/rss21/p010.html.

Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Xindong He, Xu Huang, et al. AgiBot World Colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025a.

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 *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025b. doi: 10.15607/RSS.2025.XXI.014. URL https://www.roboticsproceedings.org/rss21/p014.html.

Build AI. Egocentric-100k. Hugging Face dataset, 2025a. URL https://huggingface.co/datasets/builddotai/Egocentric-100K.

Build AI. Egocentric-10k. Hugging Face dataset, 2025b. URL https://huggingface.co/datasets/builddotai/Egocentric-10K.

Junhao Cai, Zetao Cai, Jiafei Cao, Yilun Chen, Zeyu He, Lei Jiang, Hang Li, Hengjie Li, Yang Li, Yufei Liu, Yanan Lu, Qi Lv, Haoxiang Ma, Jiangmiao Pang, Yu Qiao, Zherui Qiu, Yanqing Shen, Xu Shi, Yang Tian, Bolun Wang, Hanqing Wang, Jiaheng Wang, Tai Wang, Xueyuan Wei, Chao Wu, Yiman Xie, Boyang Xing, Yuqiang Yang, Yuyin Yang, Qiaojun Yu, Feng Yuan, Jia Zeng, Jingjing Zhang, Shenghan Zhang, Shi Zhang, Zhuoma Zhaxi, Bowen Zhou, Yuanzhen Zhou, Yunsong Zhou, Hongrui Zhu, Yangkun Zhu, and Yuchen Zhu. Internvla-a1: Unifying understanding, generation and action for robotic manipulation, 2026. URL https://arxiv.org/abs/2601.02456.

Boyu Chen, Yi Chen, Lu Qiu, Jerry Bai, Yuying Ge, and Yixiao Ge. Unit: Toward a unified physical language for human-to-humanoid policy learning and world modeling, 2026a. URL https://arxiv.org/abs/2604.19734.

Siyi Chen, Minkyu Choi, Zesen Zhao, Kuan Han, Qing Qu, and Zhongming Liu. Enhancing video representation learning with temporal differentiation. In *Conference on Parsimony and Learning*, volume 280 of *Proceedings of Machine Learning Research*, pp. 1007–1034. PMLR, 2025a. URL https://proceedings.mlr.press/v280/chen25c.html.

Tianxing Chen, Yue Chen, Zixuan Li, Junyuan Tang, Kailun Su, Haoran Lu, Weijie Wan, Baijun Chen, Songling Liu, Haowen Yan, Honghao Su, Zhiyang Dou, Kaixuan Wang, Dandan Zhang, Yunze Liu, Yan Qin, Qiwei Liang, Qiwei Wu, Zijian Lin, Wenwei Lin, Yuran Wang, Minghua He, Tianshu Wu, Ruihai Wu, Jingquan Zhou, Kai-Chong Lei, Haibao Yu, Yuanfeng Ji, Weiyang Jin, Guanyu Lin, Xiaofan Li, Qi Xiong, Renjing Xu, Zhongyu Li, Wenhao Chai, Enze Xie, Ziwei Wang, Yao Mu, Hao Dong, Wojciech Matusik, Mingyu Ding, Wenbo Ding, Ping Luo, and Masayoshi Tomizuka. RoboDojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies. *arXiv preprint arXiv:2607.04434*, 2026b. doi: 10.48550/arXiv.2607.04434. URL https://arxiv.org/abs/2607.04434.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026c. URL https://icml.cc/virtual/2026/poster/62192.

Xiaoyu Chen, Hangxing Wei, Pushi Zhang, Chuheng Zhang, Kaixin Wang, Yanjiang Guo, Rushuai Yang, Yucen Wang, Xinquan Xiao, Li Zhao, Jianyu Chen, and Jiang Bian. villa-x: Enhancing latent action modeling in vision-language-action models. In *International Conference on Learning Representations*, volume 2026, pp. 70673–70703, 2026d. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/73471f46f69a13cede6cf2db3a6f438f-Abstract-Conference.html.

Yi Chen, Yuying Ge, Weiliang Tang, Yizhuo Li, Yixiao Ge, Mingyu Ding, Ying Shan, and Xihui Liu. Moto: Latent motion token as the bridging language for learning robot manipulation from videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19752–19763, October 2025b. URL https://openaccess.thecvf.com/content/ICCV2025/html/Chen_Moto_Latent_Motion_Token_as_the_Bridging_Language_for_Learning_ICCV_2025_paper.html.

Yi Chen, Yuying Ge, Hui Zhou, Mingyu Ding, Yixiao Ge, and Xihui Liu. Dial: Decoupling intent and action via latent world modeling for end-to-end vla, 2026e. URL https://arxiv.org/abs/2603.29844.

Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 5842–5850, 2017. URL https://openaccess.thecvf.com/content_iccv_2017/html/Goyal_The_Something_Something_ICCV_2017_paper.html.

Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18995–19012, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Grauman_Ego4D_Around_the_World_in_3000_Hours_of_Egocentric_Video_CVPR_2022_paper.html.

Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, et al. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19383–19400, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Grauman_Ego-Exo4D_Understanding_Skilled_Human_Activity_from_First-_and_Third-Person_Perspectives_CVPR_2024_paper.html.

Ryan Hoque, Peide Huang, David Yoon, Mouli Sivapurapu, and Jian Zhang. Egodex: Learning dexterous manipulation from large-scale egocentric video. In *International Conference on Learning Representations*, volume 2026, pp. 4218–4237, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/07fcc6e2b89439d3ee5ab60939aaa6a0-Abstract-Conference.html.

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. doi: 10.1109/ICRA55743.2025.11127989. URL https://ieeexplore.ieee.org/document/11127989/.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Donovon Jackson, Charlotte Le, Yunshuang Li, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta,

Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. DROID: A large-scale in-the-wild robot manipulation dataset. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.120. URL https://www.roboticsproceedings.org/rss20/p120.html.

Dongyoung Kim, Huiwon Jang, Myungkyu Koo, Suhyeok Jang, Taeyoung Kim, Beomjun Kim, Byungjun Yoon, Changsung Jang, Daewon Choi, Dongsu Han, Donguk Lee, Heeseung Kwon, Hojin Jeon, Jaehyun Kang, Jaekyoung Bae, Jihyuk Lee, Jimin Lee, John Won, Joonwoo Ahn, Junhyeong Park, Junyoung Sung, Kyungmin Lee, Minseong Han, Minsung Yoon, Sejune Joo, Seonil Son, Seungcheol Park, Seunggeun Cho, Seungjun Moon, Seungku Kim, Yonghoon Dong, Yongjin Cho, Youngchan Kim, Chang Hwan Kim, Dohyeon Kim, Heecheol Kim, Heewon Lee, Hensen Ahn, Hyungkyu Ryu, Hyunsoo Choi, Hyunsoo Shin, Jaeheon Jung, Jaewoo Kim, Jinwook Kim, Joochul Chang, Joonsoo Kim, Junghun Park, Jungwoo Park, Junho Cho, Junhyeok Park, Junwon Lee, Kangwook Lee, Kwanghoon Kim, Kyoungwhan Choe, Manoj Bhadu, Nayoung Oh, Sangjun Kim, Sangwoo Kim, Seunghoon Shim, Seunghyun Kim, Seungjun Lee, Seungyup Ka, Sungryol Yang, Wook Jung, Yashu Shukla, Yeonjae Lee, Yeonwoo Bae, and Jinwoo Shin. Rldx-1 technical report, 2026. URL https://arxiv.org/abs/2605.03269.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Luyao Zhang, Mingrui Yu, Zelin Gao, Nan Xue, Boyu Zhou, Xing Zhu, Mingyu Ding, Yujun Shen, and Yinghao Xu. Causal world modeling for robot control. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026a. doi: 10.15607/RSS.2026.XXII.016. URL https://www.roboticsproceedings.org/rss22/p016.html.

Yixuan Li, Yuhui Chen, Mingcai Zhou, and Haoran Li. QDepth-VLA: Quantized depth prediction as auxiliary supervision for vision-language-action models. In *Proceedings of the 25th International Conference on Autonomous Agents and Multiagent Systems*, pp. 2771–2780. International Foundation for Autonomous Agents and Multiagent Systems, 2026b. doi: 10.65109/LJRK3716. URL https://ifaamas.csc.liv.ac.uk/Proceedings/aamas2026/pdfs/LJRK3716.pdf.

Hao Luo, Yicheng Feng, Wanpeng Zhang, Sipeng Zheng, Ye Wang, Haoqi Yuan, Jiazheng Liu, Chaoyi Xu, Haiweng Xu, Qin Jin, and Zongqing Lu. Vision-language-action pretraining from large-scale human videos. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026a. URL https://icml.cc/virtual/2026/poster/62813.

Hao Luo, Ye Wang, Wanpeng Zhang, Haoqi Yuan, Yicheng Feng, Haiweng Xu, Sipeng Zheng, and Zongqing Lu. Joint-aligned latent action: Towards scalable VLA pretraining in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 35047–35058, June 2026b. URL https://openaccess.thecvf.com/content/CVPR2026/html/Luo_Joint-Aligned_Latent_Action_Towards_Scalable_VLA_Pretraining_in_the_Wild_CVPR_2026_paper.html.

Jiangran Lyu, Kai Liu, Xuheng Zhang, Haoran Liao, Yusen Feng, Wenxuan Zhu, Tingrui Shen, Jiayi Chen, Jiazhao Zhang, Yifei Dong, Wenbo Cui, Senmao Qi, Shuo Wang, Yixin Zheng, Mi Yan, Xuesong Shi, Haoran Li, Dongbin Zhao, Ming-Yu Liu, Zhizheng Zhang, Li Yi, Yizhou Wang, and He Wang. Lda-1b: Scaling latent dynamics action model via universal embodied data ingestion. In *Proceedings of Robotics: Science and Systems*, 2026. doi: 10.15607/RSS.2026.XXII.210. URL https://www.roboticsproceedings.org/rss22/p210.html.

Teli Ma, Jia Zheng, Zifan Wang, Chunli Jiang, Andy Cui, Junwei Liang, and Shuo Yang. Dit4dit: Jointly modeling video dynamics and actions for generalizable robot control, 2026. URL https://arxiv.org/abs/2603.10448.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of household tasks for generalist robots. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.050. URL https://www.roboticsproceedings.org/rss20/p050.html.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://research.nvidia.com/labs/lpr/publication/gr00tn1_2025/.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11671. URL https://ojs.aaai.org/index.php/AAAI/article/view/11671.

Ryan Punamiya, Simar Kareer, Zeyi Liu, Joshua Citron, Ri-Zhao Qiu, Xiongyi Cai, Alexey Gavryushin, Jiaqi Chen, Davide Liconti, Lawrence Y. Zhu, Patcharapong Aphiwetsa, Baoyu Li, Aniketh Cheluva, Pranav Kuppili, Yangcen Liu, Dhruv Patel, Aidan Gao, Ryan Co, Hye-Young Chung, Renee Zbizika, Jinyun Liu, Xiaomeng Xu, Haoyu Xiong, Geng Chen, Sebastiano Oliani, Wenkai Xuan, Chenyu Yang, Xi Wang, James Fort, Richard Newcombe, Josh Gao, Jason Chong, Garrett Matsuda, Aseem Doriwala, Robert K. Katzschmann, Marc Pollefeys, Xiaolong Wang, Shuran Song, Judy Hoffman, and Danfei Xu. Egoverse: An egocentric human dataset for robot learning from around the world. In *Proceedings of Robotics: Science and Systems*, 2026. doi: 10.15607/RSS.2026.XXII.092. URL https://www.roboticsproceedings.org/rss22/p092.html.

Ropedia. Xperience-10m: A large-scale egocentric multimodal dataset with structured 3d/4d annotations. Hugging Face dataset, 2026. URL https://huggingface.co/datasets/ropedia-ai/xperience-10m.

Sandeep Kumar Routray, Hengkai Pan, Unnat Jain, Shikhar Bahl, and Deepak Pathak. ViPRA: Video prediction for robot actions. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/707e34efcabd2b9375f7a64019600aa8-Abstract-Conference.html.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=2NlGyqNjns.

Himanshu Gaurav Singh, Antonio Loquercio, Carmelo Sferrazza, Jane Wu, Haozhi Qi, Pieter Abbeel, and Jitendra Malik. Hand-object interaction pretraining from videos. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3352–3360. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11127811. URL https://ieeexplore.ieee.org/document/11127811/.

StarVLA Community. Starvla: A lego-like codebase for vision-language-action model developing, 2026. URL https://arxiv.org/abs/2604.05014.

Wan-AI. Wan2.2-ti2v-5b. Hugging Face model card, 2025. URL https://huggingface.co/Wan-AI/Wan2.2-TI2V-5B.

Limin Wang, Zhan Tong, Bin Ji, and Gangshan Wu. TDN: Temporal difference networks for efficient action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision*

and Pattern Recognition (CVPR), pp. 1895–1904, June 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Wang_TDN_Temporal_Difference_Networks_for_Efficient_Action_Recognition_CVPR_2021_paper.html.

Shihan Wu, Xuecheng Liu, Shaoxuan Xie, Pengwei Wang, Xinghang Li, Bowen Yang, Zhe Li, Kai Zhu, Hongyu Wu, Yiheng Liu, Zhaoye Long, Runtian Xu, Yue Wang, Chong Liu, Dihan Wang, Ziqiang Ni, Xiang Yang, You Liu, Ruoxuan Feng, Lei Zhang, Denghang Huang, Chenghao Jin, Anlan Yin, Xinlong Wang, Zhenguo Sun, Junkai Zhao, Mengfei Du, Mingyu Cao, Xiansheng Chen, Hongyang Cheng, Xiaojie Zhang, Yankai Fu, Ning Chen, Cheng Chi, Sixiang Chen, Huaihai Lyu, Xiaoshuai Hao, Yequan Wang, Bo Lei, Dong Liu, Xi Yang, Yance Jiao, Tengfei Pan, Yunyan Zhang, Songjing Wang, Ziqian Zhang, Xu Liu, Ji Zhang, Caowei Meng, Zhizheng Zhang, Jiyang Gao, Song Wang, Xiaokun Leng, Zhiqiang Xie, Zhenzhen Zhou, Peng Huang, Wu Yang, Yandong Guo, Yichao Zhu, Suibing Zheng, Hao Cheng, Xinmin Ding, Yang Yue, Huanqian Wang, Chi Chen, Jingrui Pang, YuXi Qian, Haoran Geng, Lianli Gao, Haiyuan Li, Bin Fang, Gao Huang, Yaodong Yang, Hao Dong, He Wang, Hang Zhao, Yadong Mu, Di Hu, Hao Zhao, Tiejun Huang, Shanghang Zhang, Yonghua Lin, Zhongyuan Wang, and Guocai Yao. RoboCOIN: An open-sourced bimanual robotic data COllection for INtegrated manipulation, 2025. URL https://arxiv.org/abs/2511.17441.

Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10371–10381, June 2024a. URL https://openaccess.thecvf.com/content/CVPR2024/html/Yang_Depth_Anything_Unleashing_the_Power_of_Large-Scale_Unlabeled_Data_CVPR_2024_paper.html.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. In *Advances in Neural Information Processing Systems*, volume 37, pp. 21875–21911. Curran Associates, Inc., 2024b. doi: 10.52202/079017-0688. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/26cfdcd8fe6fd75cc53e92963a656c58-Abstract-Conference.html.

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, 2025. URL https://arxiv.org/abs/2507.12440.

Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Dekang Qi, Junjin Xiao, Haoyun Liu, Ronghan Chen, Yuzhi Chen, Dongjie Huo, Feng Xiong, Xing Wei, Zhiheng Ma, and Mu Xu. Abot-m0: Vla foundation model for robotic manipulation with action manifold learning, 2026. URL https://arxiv.org/abs/2602.11236.

Jinhui Ye, Ning Gao, Senqiao Yang, Jinliang Zheng, Zixuan Wang, Yuxin Chen, Pengguang Chen, Yilun Chen, Shu Liu, and Jiaya Jia. Starvla-α: Reducing complexity in vision-language-action systems, 2026a. URL https://arxiv.org/abs/2604.11757.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, Lars Liden, Kimin Lee, Jianfeng Gao, Luke Zettlemoyer, Dieter Fox, and Minjoon Seo. Latent action pretraining from videos. In *International Conference on Learning Representations*, volume 2025, pp. 28213–28239, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/45d74e190008c7bff2845ffc8e3facd3-Abstract-Conference.html.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Malik, Kyungmin Lee, William Liang, Nadun Ranawaka, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Ryan Julian, Danfei Xu, Yilun Du, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi "Jim" Fan, and Joel Jang. World action models are zero-shot policies, 2026b. URL https://arxiv.org/abs/2602.15922.

Bin Yu, Shijie Lian, Xiaopeng Lin, Zhaolong Shen, Yuliang Wei, Changti Wu, Hang Yuan, Haishan Liu, Bailing Wang, Cong Huang, and Kai Chen. Frameskip: Learning from fewer but more informative frames in vla training, 2026. URL https://arxiv.org/abs/2605.13757.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination?, 2026. URL https://arxiv.org/abs/2603.16666.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Tai Wang, Ya-Qin Zhang, Jingjing Liu, and Xianyuan Zhan. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *International Conference on Learning Representations*, volume 2026, pp. 60580–60606, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/630d293833e09e1ecd892a898a20b074-Abstract-Conference.html.

A Additional Experimental Details

A.1 Latent Action Pretraining Data

The LAM is pretrained for 400,000 optimization steps using a batch size of 1,024. Table 6 summarizes the video data used for pretraining. Each video sample contains one current frame and four evenly sampled future frames. For normal-speed videos, the sampled clip spans approximately 1.6 seconds. For some human videos with fast hand motions, we use shorter windows to avoid covering excessive manipulation progress within a single clip. This temporal adaptation applies to the LAM pretraining corpus; sampling during robot policy training is described in Appendix A.2.

The RoboCasa and RoboTwin videos used for LAM pretraining come exclusively from their training sets. No evaluation rollouts or test-set videos are included, and pretraining does not use action annotations. In the 10% robot-demonstration setting, the LAM has already seen videos from the full training set. Thus, 10% denotes the action-supervision budget during policy training, not the amount of video available to LAM pretraining.

Table 6: Composition of the video corpus used to pretrain the semantic-geometric latent action model.

Data category	Dataset	Hours
Human interaction videos	SSv2 (Goyal et al., 2017)	240
	Ego-Exo4D (Grauman et al., 2024)	1,286
	Xperience-10M (Ropedia, 2026)	921
	Egocentric-10K (Build AI, 2025b)	10,000
	Ego4D (Grauman et al., 2022)	3,670
	EgoVerse (Punamiya et al., 2026)	1,362
	EgoDex (Hoque et al., 2026)	830
	Subtotal	*18,309*
Real-robot data	RoboCOIN (Wu et al., 2025)	704
	AgiBot World 2026 (AgiBot World Team, 2026)	433
	AgiBot World Beta (Bu et al., 2025a)	2,007
	Subtotal	*3,144*
Simulated robot data	InternData-A1 (Cai et al., 2026)	3,656
	RoboCasa (Nasiriany et al., 2024)	81
	RoboTwin (Chen et al., 2026c)	34
	Subtotal	*3,771*
Total		**25,224**

A.2 Policy Inputs and Video Sampling

Observations and instructions. The policy normally receives a primary view and two wrist views, whereas the LAM uses only the primary view. For egocentric human videos, the two missing wrist views are represented by learnable embeddings, with text indicating which views are missing. Robot state is omitted when unavailable. For each task scene in our self-collected human videos, a vision-language model generates 20 semantically equivalent instructions. Human videos and the corresponding robot demonstrations share this instruction set, from which an instruction is sampled randomly during training. These videos have no action annotations; language instructions provide task context, not motion targets.

Temporal sampling. Real-robot policy training uses our self-collected human videos. We match the manipulation pace to robot demonstrations during recording, or adjust the video speed afterward when human videos are collected at a faster pace. Human and robot clips are then sampled over the same temporal duration, corresponding to one robot action chunk. Each clip contains one current frame and four future frames at evenly spaced offsets. This duration matching does not require frame-aligned human and robot demonstrations.

Mixed policy training. When co-training with human videos and robot demonstrations, the two data sources are sampled with weights of 0.5 and 0.5, respectively. These are sampling weights rather than a fixed equal count in every minibatch. Encoder distillation and decoder prediction apply to both sources, while the action loss applies only to samples with robot action annotations. For RoboCasa video scaling, additional robot videos are used without action supervision and receive both LAM losses. The loss reductions are specified in Appendix A.4.

A.3 Model and Training Configuration

Tables 7 and 8 summarize the architecture and optimization settings. WALA and WALA-Wan share the semantic-geometric LAM and its training losses; their policy-backbone interfaces are compared in Fig. 7.

Latent action adapters. The encoder and decoder adapters have the same architecture. Each first projects the H policy tokens to dimension 1,024 and adds learned temporal embeddings. A one-layer, eight-head Transformer decoder uses 32 learned queries to attend to these tokens and returns 32 tokens of dimension 1,024. Thus, each adapter maps $\mathbb{R}^{H\times d_{\text{policy}}}$ to $\mathbb{R}^{32\times 1024}$, where d_{policy} is the policy token dimension.

Table 7: Core model settings for WALA and WALA-Wan on RoboTwin.

Policy-backbone settings

Component	Standard WALA	WALA-Wan
Policy backbone	Qwen3-VL-4B-Instruct	Wan2.2-TI2V-5B
Policy input resolution	256×256	384×320
Action head	MLP (hidden size 2,560)	MLP (hidden size 3,072)
Latent action generation	Learned action queries in the policy backbone	2-layer, 24-head action-query resampler over backbone hidden tokens

Shared semantic-geometric latent action model

Component	Configuration
Semantic feature encoder	DINOv3 ViT-B/16; feature dimension 768
Depth estimator	Depth Anything V2-L; input resolution 224×224
Latent action representation	32 latent action tokens; hidden dimension 1,024
LAM encoder and decoder	8 encoder layers; 4 decoder layers; 8 attention heads

The essential optimization settings are summarized in Table 8. Batch size denotes the effective total batch size across devices. All stages use AdamW with $(\beta_1, \beta_2) = (0.9, 0.95)$ and $\epsilon = 10^{-8}$, cosine learning-rate decay with a minimum learning rate of 5×10^{-7}, gradient clipping at 1.0, and random seed 42 unless otherwise specified.

Table 8: Essential training settings for LAM pretraining and policy training.

Setting	LAM pretraining	RoboTwin	RoboCasa	Real robot
Training steps	400,000	200,000	100,000	50,000
Effective batch size	1,024	192	256	256
Action dimension	–	14	47	14
Action horizon H	–	50	32	50
Default learning rate	1×10^{-4}	1×10^{-5}	3×10^{-5}	1×10^{-5}
Policy-backbone learning rate	–	1×10^{-5}	1×10^{-5}	1×10^{-5}
Decoder learning rate	1×10^{-4}	1×10^{-5}	1×10^{-5}	1×10^{-5}
Action head / adapters learning rate	–	1×10^{-4}	1×10^{-4}	1×10^{-4}
Robot-state dropout	–	0	0	0.1

For WALA-Wan on RoboTwin, we use the same effective batch size of 192 and 200,000 training steps as standard WALA. The Wan backbone uses a learning rate of 1×10^{-5}, while the action-query resampler uses 1×10^{-4}. The Wan VAE and text encoder remain frozen during training.

During Stage I, the trainable LAM modules are initialized from scratch, while DINOv3 and Depth Anything V2 remain frozen. Standard policy training on RoboTwin and RoboCasa-GR1-Tabletop uses the pretrained LAM. Real-robot policy training is initialized from a complete RoboTwin-trained WALA checkpoint to accelerate convergence; this initialization is not large-scale robot-policy pretraining.

Ablation configurations. All variants in Table 3 retain robot action supervision. Removing distillation or prediction supervision leaves the remaining loss weights unchanged. The semantic-only variant uses a LAM whose encoder does not encode depth and whose decoder does not predict depth. The semantic-only and semantic-geometric LAMs have different architectures and are pretrained in separate runs. The variant without LAM pretraining initializes its decoder randomly, while retaining the pretrained policy backbone and frozen visual feature extractors.

Training and policy inference for RoboTwin and RoboCasa-GR1-Tabletop are performed on Alibaba Cloud's PPU platform. NVIDIA H20 GPUs are used for RoboTwin environment rendering.

LAM pretraining resources and reuse. The semantic-geometric LAM contains approximately 400 million parameters. Pretraining on the 25,224-hour video corpus for 400,000 optimization steps with a global batch size of 1,024 takes approximately 200 hours on 128 Alibaba Cloud PPU accelerators, corresponding to approximately 25,600 PPU-hours for this pretraining run. The pretrained LAM is reused across downstream tasks without repeating large-scale video pretraining. During robot policy training, task-relevant egocentric human videos provide additional supervision without action annotations, improving manipulation performance with limited robot demonstrations.

LAM-based baseline reproductions on RoboCasa. We adapt the released Moto (Chen et al., 2025b) and UniVLA (Bu et al., 2025b) implementations to RoboCasa-GR1-Tabletop. For a fair comparison on the target benchmarks, the Moto and UniVLA results in Table 2 use LAM checkpoints fine-tuned on RoboTwin and RoboCasa. These checkpoints are also used for the action probes and encoder-attention visualizations. Both use the same training set as WALA, with 1,000 demonstrations for each of the 24 tasks, a global batch size of 256, a training budget of 100,000 optimizer updates, and 32-step action chunks. Each baseline is trained on 16 Alibaba Cloud PPU accelerators, using the same accelerator platform as WALA. Evaluation follows the same benchmark protocol as WALA, with 50 episodes per task, 32 actions executed per policy prediction, a limit of 720 simulator steps per episode, and the same simulator-defined success criterion. Moto and UniVLA achieve 547/1200 (45.58%) and 614/1200 (51.17%), respectively. Results are rounded to one decimal place in Table 2.

A.4 Loss Implementation

The regression notation in the Method distinguishes the semantic and geometric terms without prescribing the same penalty for both. Semantic-feature regression uses L1, whereas depth-map regression uses Smooth L1,

$$\begin{aligned} \mathcal{L}_{\text{reg}}^{\text{sem}} &= \text{mean}\left|\widehat{\Delta X} - \Delta X\right|, \\ \mathcal{L}_{\text{reg}}^{\text{dep}} &= \text{SmoothL1}(\widehat{\Delta D}, \Delta D). \end{aligned} \tag{12}$$

The semantic cosine term $\mathcal{L}_{\cos}$ is one minus the cosine similarity between predicted and target feature-delta vectors, averaged over samples, future steps, and spatial locations. It complements element-wise regression by matching the direction of feature change.

The depth-gradient term compares horizontal and vertical finite differences,

$$\mathcal{L}_{\text{grad}} = \sum_{d \in \{x,y\}} \text{mean}\left|\nabla_d \widehat{\Delta D} - \nabla_d \Delta D\right|. \tag{13}$$

This term encourages more accurate recovery of local depth variations. All four prediction terms are evaluated in delta space. The depth maps used by the LAM are estimated at 224×224 resolution, and the decoder predicts depth deltas at this resolution. Before depth deltas are formed, the current and future maps share a clip-wise normalization: log-depth values are centered by their median, scaled by their median absolute deviation, and clipped to $[-3, 3]$. The depth-map notation in the

Method refers to these normalized estimates. Sharing the normalization statistics puts all frames on a common scale within the clip, but does not remove errors or temporal inconsistencies in monocular depth estimation.

For latent-action distillation in Eq. (10), let $U = (u_1, \ldots, u_{N_z})$ and $V = (v_1, \ldots, v_{N_z})$ denote the adapted policy tokens and encoder targets, respectively, with $N_z = 32$. Then

$$\ell_{\cos}(U, V) = \frac{1}{N_z} \sum_{j=1}^{N_z} \left(1 - \frac{\langle u_j, v_j \rangle}{\|u_j\|_2 \|v_j\|_2}\right). \tag{14}$$

Layer normalization is applied independently over each token's feature dimension without learned affine parameters. Smooth L1 is averaged over tokens and feature dimensions, with transition threshold 1. Encoder targets receive no gradient. We set $\lambda_{\cos} = 0.1$, $\lambda_{\text{grad}} = 0.2$, and $\lambda_{\text{dep}} = 1.0$ for LAM pretraining. The policy's world-model loss uses the same prediction terms and weights, with $\lambda_{\text{distill}} = 0.005$ and $\lambda_{\text{wm}} = 0.5$ in the overall policy objective.

Loss-weight selection. We choose the policy-loss weights empirically based on the magnitudes of the separately averaged loss terms during training. The action-loss coefficient is fixed at 1. The auxiliary weights are selected to balance distillation and world-model supervision without overwhelming the primary action-prediction objective, and remain fixed during training. These auxiliary terms also provide representation-level regularization by constraining policy-generated latent actions through distillation and future prediction. Decoder prediction compares the policy tokens' predicted outcomes with the semantic-feature and depth-map deltas in downstream videos, rather than requiring them only to reproduce the frozen encoder's targets. The action loss jointly links the policy tokens to recorded robot actions.

Minibatch reduction. Let $\mathcal{B}$ denote a policy-training minibatch and $\mathcal{B}_{\text{lab}}$ its samples with robot action annotations. For a labeled sample i, the action loss is

$$\mathcal{L}_{\text{act},i} = \frac{1}{H d_a} \sum_{h=0}^{H-1} \left\| \widehat{a}_{i,t+h} - a_{i,t+h} \right\|_1, \tag{15}$$

where d_a is the action dimension and H is the chunk length. This loss is averaged over $\mathcal{B}_{\text{lab}}$; distillation and world-model losses are averaged over all of $\mathcal{B}$,

$$\begin{aligned} \mathcal{L}^{\mathcal{B}}_{\text{policy}} = {} & \frac{1}{\max(1, |\mathcal{B}_{\text{lab}}|)} \sum_{i \in \mathcal{B}_{\text{lab}}} \mathcal{L}_{\text{act},i} \\ & + \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \left(\lambda_{\text{distill}} \mathcal{L}_{\text{distill},i} + \lambda_{\text{wm}} \mathcal{L}_{\text{wm},i}\right). \end{aligned} \tag{16}$$

The action term is zero for an unlabeled minibatch. This separately normalized reduction implements the masked objective in Sec. 3.3 without including unlabeled samples in the action-loss denominator.

Visualization. Figure 2 displays the decoder's predicted signed deltas. Neutral gray indicates little change; DINOv3 delta colors visualize feature directions through PCA rather than fixed semantic categories. Depth-delta brightness indicates signed changes in normalized depth estimates, not metric distances. In Fig. 6, policy-generated latent action tokens condition the LAM decoder through the decoder adapter. Adding its predicted deltas to the current DINOv3 features and depth maps yields the future predictions shown alongside their targets. The DINOv3 feature maps are PCA-projected and overlaid on RGB frames, while depth maps are shown directly.

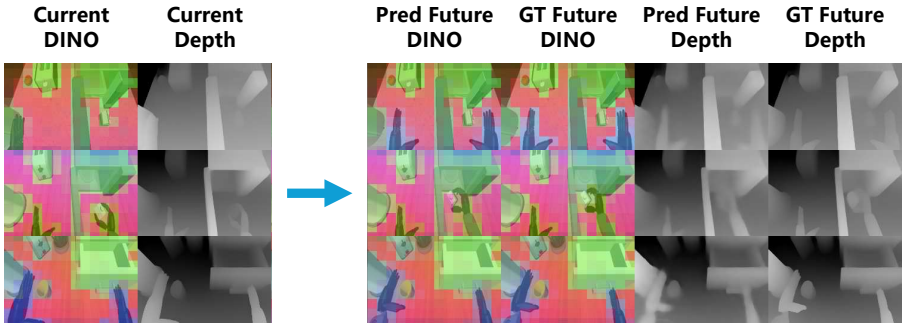

Figure 6: Future prediction from policy-generated latent action tokens. Through the decoder adapter, these tokens condition the LAM decoder to predict semantic-feature and depth-map deltas. Adding the deltas to the current representations yields the future predictions shown alongside their targets. DINOv3 features are PCA-projected to RGB and overlaid on the corresponding RGB frames; depth maps are shown directly.

A.5 Policy Backbone Variants

Figure 7 compares the action-query interfaces of the two backbones specified in Sec. 4.1. Both use the training objective in Sec. 3.3.

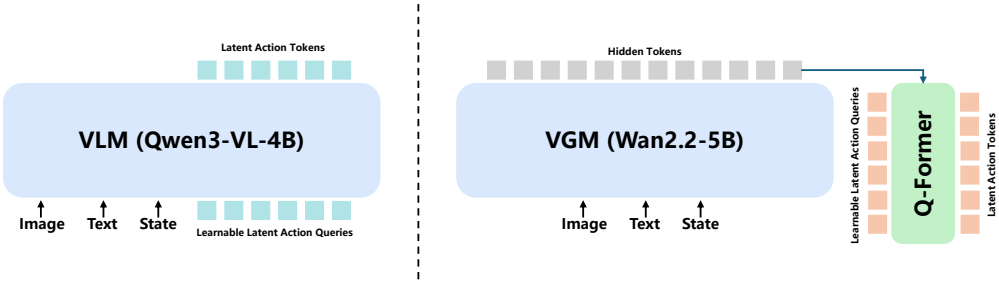

Figure 7: Comparison of the two WALA policy backbones. Standard WALA reads latent action tokens from learned action queries in Qwen3-VL-4B-Instruct, while WALA-Wan uses an action-query resampler over Wan2.2-TI2V-5B hidden tokens. Both variants use the same latent-action distillation, semantic-geometric world-model supervision, and robot-action prediction objectives.

A.6 Real-Robot Experimental Protocol

For base-task comparisons, WALA, π_0, and $\pi_{0.5}$ each jointly train a single policy on only the same 4 base tasks. The baselines use the same 200 robot demonstrations per base task and evaluation protocol as WALA's 200-demonstration setting.

For transfer experiments, each WALA data setting trains a single policy in a joint 10-task setting comprising the 4 base tasks and 6 transfer tasks, using the available robot demonstrations and human videos. In both transfer conditions, each base task uses 200 robot demonstrations and 400 human videos; only the target-task human-video budget varies. In the condition without target-task human videos, the 3 zero-shot tasks have neither robot demonstrations nor human videos and therefore contribute no training samples. All tasks are evaluated for 30 trials per task (Tables 4 and 5). Average base-task success is computed over 120 trials.

The 3 zero-shot tasks use no target-task robot demonstrations, whereas the 3 few-shot tasks use 5 per task; both compare zero versus 150 human videos per target task. Place Two Apples on Plate requires picking and placing the apples sequentially. The 2 door-opening tasks use different cabinet instances and scene layouts. Human recordings contain only single-camera egocentric video, without measured or post-processed end-effector (EEF) trajectories.

Figure 8 visualizes representative rollouts for all 6 transfer tasks.

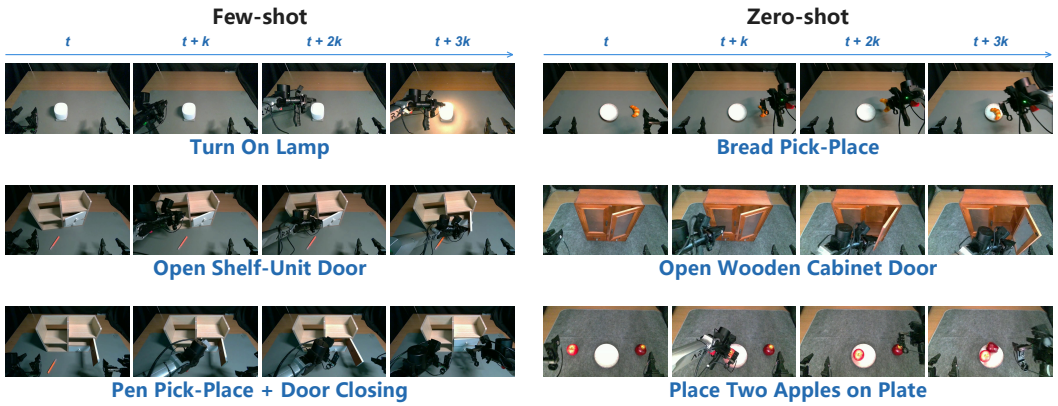

Figure 8: Real-robot transfer rollouts on the 3 few-shot tasks (left) and 3 zero-shot tasks (right). Each row shows 4 equally spaced frames spanning the start and completion of an operation.

A.7 Per-Task Benchmark Results

Tables 9 and 10 report individual task results. RoboTwin uses 100 evaluation episodes per task in each setting, and RoboCasa-GR1-Tabletop uses 50 per task. Within each benchmark, equal trial counts make the aggregate success rate equivalent to the unweighted mean of per-task success rates.

Table 9: Per-task RoboTwin results for WALA and WALA-Wan, reported as successful trials out of 100 evaluation episodes.

Task	WALA		WALA-Wan	
	Clean	Random	Clean	Random
adjust_bottle	100/100	100/100	100/100	100/100
beat_block_hammer	98/100	96/100	98/100	96/100
blocks_ranking_rgb	99/100	98/100	100/100	98/100
blocks_ranking_size	77/100	89/100	89/100	89/100
click_alarmclock	100/100	100/100	97/100	95/100
click_bell	100/100	99/100	98/100	96/100
dump_bin_bigbin	95/100	99/100	99/100	97/100
grab_roller	100/100	100/100	100/100	100/100
handover_block	86/100	88/100	92/100	79/100
handover_mic	100/100	100/100	98/100	100/100
hanging_mug	59/100	53/100	54/100	49/100
lift_pot	100/100	100/100	100/100	100/100
move_can_pot	84/100	87/100	87/100	90/100
move_pillbottle_pad	100/100	100/100	97/100	99/100
move_playingcard_away	100/100	100/100	99/100	98/100
move_stapler_pad	59/100	63/100	84/100	77/100
open_laptop	100/100	100/100	98/100	100/100
open_microwave	89/100	99/100	88/100	94/100
pick_diverse_bottles	67/100	80/100	76/100	72/100
pick_dual_bottles	91/100	93/100	90/100	88/100
place_a2b_left	91/100	99/100	93/100	94/100
place_a2b_right	94/100	95/100	96/100	94/100
place_bread_basket	96/100	93/100	95/100	92/100
place_bread_skillet	97/100	89/100	88/100	93/100
place_burger_fries	98/100	100/100	99/100	99/100
place_can_basket	84/100	79/100	80/100	83/100
place_cans_plasticbox	98/100	100/100	98/100	99/100
place_container_plate	100/100	100/100	99/100	97/100
place_dual_shoes	81/100	82/100	83/100	96/100
place_empty_cup	99/100	100/100	99/100	100/100
place_fan	91/100	96/100	98/100	94/100
place_mouse_pad	74/100	85/100	93/100	88/100
place_object_basket	90/100	87/100	90/100	87/100
place_object_scale	97/100	100/100	96/100	99/100
place_object_stand	98/100	98/100	91/100	94/100
place_phone_stand	96/100	99/100	98/100	95/100
place_shoe	99/100	100/100	98/100	99/100
press_stapler	99/100	100/100	97/100	99/100
put_bottles_dustbin	73/100	82/100	87/100	85/100
put_object_cabinet	89/100	90/100	89/100	100/100
rotate_qrcode	84/100	82/100	83/100	87/100
scan_object	88/100	93/100	92/100	89/100
shake_bottle	100/100	100/100	100/100	99/100
shake_bottle_horizontally	100/100	100/100	100/100	99/100
stack_blocks_three	94/100	98/100	98/100	99/100
stack_blocks_two	100/100	100/100	100/100	99/100
stack_bowls_three	84/100	85/100	84/100	85/100
stack_bowls_two	96/100	100/100	98/100	91/100
stamp_seal	88/100	88/100	82/100	92/100
turn_switch	49/100	76/100	70/100	71/100

WALA overall: 4531/5000 (90.62%) on Clean and 4640/5000 (92.80%) on Random; 9171/10000 (91.71%) combined.
WALA-Wan overall: 4618/5000 (92.36%) on Clean and 4615/5000 (92.30%) on Random; 9233/10000 (92.33%) combined.

Table 10: Per-task WALA results on RoboCasa-GR1-Tabletop. Results show successful trials out of 50 evaluation episodes and the corresponding success rates.

#	Task	Success	Rate	#	Task	Success	Rate
1	PnPBottleToCabinetClose	41/50	82.0%	13	PlacematToBowl	37/50	74.0%
2	PnPCanToDrawerClose	48/50	96.0%	14	PlacematToPlate	33/50	66.0%
3	PnPCupToDrawerClose	43/50	86.0%	15	PlacematToTieredshelf	21/50	42.0%
4	PnPMilkToMicrowaveClose	39/50	78.0%	16	PlateToBowl	36/50	72.0%
5	PnPPotatoToMicrowaveClose	39/50	78.0%	17	PlateToCardboardbox	25/50	50.0%
6	PnPWineToCabinetClose	31/50	62.0%	18	PlateToPan	26/50	52.0%
7	CuttingboardToBasket	43/50	86.0%	19	PlateToPlate	46/50	92.0%
8	CuttingboardToCardboardbox	33/50	66.0%	20	TrayToCardboardbox	43/50	86.0%
9	CuttingboardToPan	47/50	94.0%	21	TrayToPlate	49/50	98.0%
10	CuttingboardToPot	40/50	80.0%	22	TrayToPot	42/50	84.0%
11	CuttingboardToTieredbasket	25/50	50.0%	23	TrayToTieredbasket	40/50	80.0%
12	PlacematToBasket	48/50	96.0%	24	TrayToTieredshelf	27/50	54.0%

Overall: 902/1200 (75.17%).

B Additional Analyses of Latent Actions

B.1 Latent Action Retrieval

Figure 9 illustrates retrieval using latent actions encoded by the pretrained LAM, recovering similar manipulation primitives across different objects, layouts, and backgrounds.

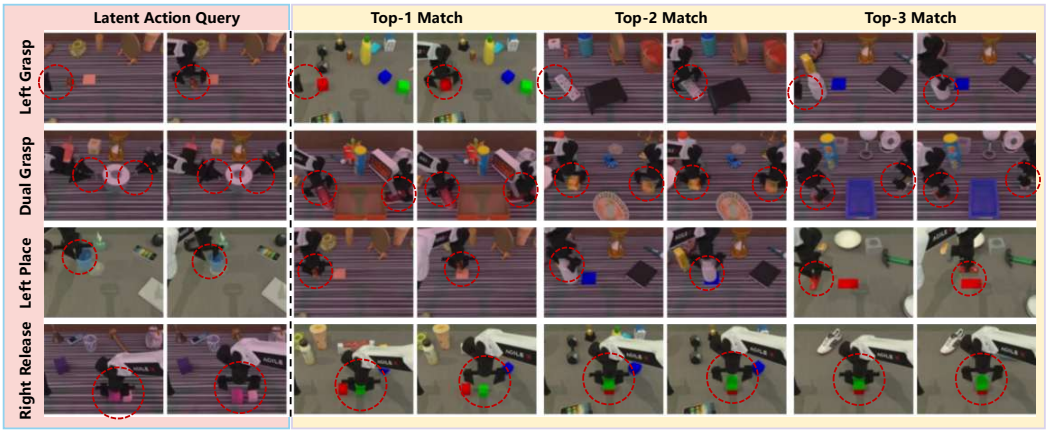

Figure 9: Retrieval using LAM-encoded latent actions. Example queries retrieve similar manipulation primitives across different objects, layouts, and backgrounds.

B.2 Decoder Dependence on Policy-Generated Tokens

Protocol. We examine whether the semantic-geometric decoder uses the latent action tokens produced by the RoboCasa policy. We freeze the jointly trained policy and decoder for this diagnostic. We select 240 clips, one from each of ten demonstration episodes for each of the 24 tasks, and predict semantic-feature and depth-map deltas at future frame offsets 8, 16, 24, and 32. This is an offline diagnostic on the benchmark's training-distribution demonstrations. For each clip, the current visual features and normalized depth are held fixed. We keep the original tokens, replace them with tokens from a different episode of the same task or from another task, or set them to zero. Replacements are made after the decoder adapter and preserve the token sequence shape. Each swap condition averages three independently sampled donors per recipient; these are repeated interventions on one trained model, not independently trained policies.

Table 11: Decoder token replacements on RoboCasa-GR1-Tabletop. Errors are averaged over 240 clips; lower is better. The last column reports the relative increase in the world-model loss compared with original tokens.

Decoder tokens	Semantic delta MSE ↓	Depth delta MSE ↓	$\mathcal{L}_{\text{wm}}$ ↓	Loss increase
Original	0.0142	0.0182	0.1392	—
Same-task swap	0.0310	0.4646	0.4036	+189.9%
Cross-task swap	0.0341	0.6394	0.4838	+247.6%
Zero tokens	0.0289	0.4020	0.3907	+180.7%

Results. Table 11 reports MSE for the predicted DINOv3 feature deltas and depth-map deltas, together with the world-model loss defined in Appendix A.4, before multiplication by its outer policy-loss weight. Both same-task and cross-task swaps increase prediction error. After averaging the three donors, the world-model loss increases for all 240 recipient clips under either swap. Same-task replacement already raises this loss by 189.9%, indicating that a shared task identity does not substitute for the original sample-specific tokens. Cross-task replacement produces a larger increase of 247.6%. These results show that the decoder relies on policy-generated latent actions to predict future semantic and geometric changes.

Figure 10 shows one of six examples selected before inspecting the intervention results. With the original tokens, the predicted semantic changes and future geometry follow the reference motion. Same-task and cross-task replacements instead introduce changes in incorrect regions and distort the predicted geometry and temporal progression, despite an unchanged current observation. With zero tokens, the projected semantic deltas become nearly neutral gray, while the reconstructed depth changes little across future steps and largely retains the current scene layout. In this example, removing the latent action information suppresses the predicted motion, whereas replacing it with another sample's tokens produces mismatched changes.

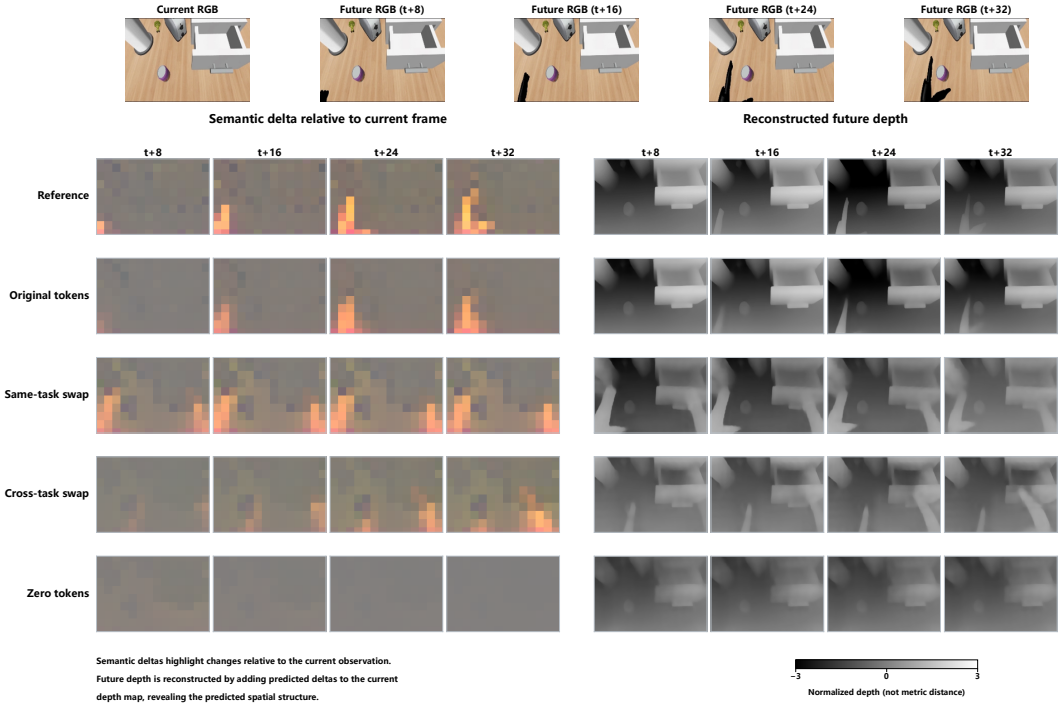

Figure 10: Decoder predictions under token replacement. Left: semantic-feature deltas relative to the current observation, displayed with a shared PCA projection and color scale across all conditions and future frames. Gray denotes zero projected change; colors do not represent fixed semantic categories. Right: future depth reconstructed by adding predicted depth deltas to the current depth map. Reference feature deltas and depth maps are derived from observed frames using DINOv3 and Depth Anything V2.

B.3 Action Probes of Frozen Latent Representations

Data and targets. We test how readily recorded robot actions can be recovered from frozen representations. The WALA LAM is taken from video pretraining, before robot-policy training. For a fair comparison on RoboTwin and RoboCasa, UniVLA (Bu et al., 2025b) and Moto (Chen et al., 2025b) use LAM checkpoints fine-tuned on RoboTwin and RoboCasa. The same checkpoints are used for their encoder-attention visualizations in Fig. 4. All LAMs remain frozen during probe feature extraction. For RoboCasa-GR1-Tabletop, we sample 100 episodes per task and split them into 80 training, 10 validation, and 10 test episodes. For RoboTwin, we use 50 Clean and 50 Random episodes per task, with a 40/5/5 split within each setting. For the official RoboDojo dataset (Chen et al., 2026b), we use 100 episodes from each of 18 real-robot and 35 simulated task groups, with an 80/10/10 split within each group. Clean and Random data are pooled for the RoboTwin probe, and real and simulated data are pooled for the RoboDojo probe. We sample up to three non-overlapping action windows per episode and repeat sampling and probe fitting with seeds 42, 43, and 44. All compared representations use the same episodes and windows within each seed. RoboCasa windows contain 32 actions and frames at offsets $[0, 8, 16, 24, 32]$; RoboTwin and RoboDojo windows contain 50 actions and frames at $[0, 12, 25, 38, 50]$. The probe predicts recorded action sequences, using arm, hand, and waist actions throughout fitting and evaluation on RoboCasa, and the bimanual action vectors on RoboTwin and RoboDojo. No language, robot state, or wrist images are supplied.

WALA's LAM pretraining includes RoboCasa and RoboTwin training videos (Appendix A.1), but neither the real nor simulated RoboDojo videos. RoboDojo therefore tests the frozen WALA representation on an unseen dataset, with a new probe fitted on its training split.

Representations and probe fitting. Current visual features concatenate DINOv3 features and normalized depth from the primary view. The visual feature sequence combines this current feature vector with four future feature deltas, $[v_t;\ \{v_{t+k} - v_t\}_{k\in\mathcal{K}}]$, where $\mathcal{K}$ is the set of four future-frame offsets and v includes both semantic and depth features.

For UniVLA and Moto, we concatenate representations from four consecutive observation pairs spanning the same five-frame clip.

Each feature group is projected by PCA using the same 4,096 training windows, retaining up to 512 nondegenerate components, followed by scaling each projected coordinate by its training standard deviation. We fit linear ridge probes with an unpenalized intercept, selecting regularization on the full action targets in validation data from $\{0, 10^{-10}, 10^{-9}, \ldots, 10^{1}\}$. Action coordinates are standardized using training-set statistics. Table 12 reports test MSE averaged over action coordinates, timesteps, and windows, followed by the mean and sample standard deviation across the three seeds. The no-visual baseline predicts the training-set mean action at every step.

Results. Combining current visual features with WALA latent actions yields lower mean error than the visual baselines and both representations of UniVLA and Moto on all three datasets. It achieves the lowest error on RoboCasa (0.541) and RoboDojo (0.308), while WALA latent actions alone perform best on RoboTwin (0.287). Without current visual features, WALA also outperforms both representations of UniVLA and Moto on all three datasets. Together, these comparisons show that WALA's semantic-geometric latent actions retain information that more readily supports linear decoding of robot actions, including on RoboDojo videos not used in WALA pretraining.

The benefit of current visual features depends on the dataset. WALA latent actions alone are competitive on RoboCasa and strongest on RoboTwin, whereas adding current visual features reduces RoboDojo error from 0.358 to 0.308. Thus, the latent actions carry substantial action-related information, while current semantic-geometric observations can provide complementary context.

For both UniVLA and Moto, continuous latent actions consistently yield lower error than their quantized counterparts, both with and without current visual features. This suggests that vector quantization may discard some action-related information that is accessible to a linear probe.

Table 12: Action decoding from frozen representations. Test MSE is measured in standardized action coordinates and reported as mean ± sample standard deviation over three seeds; lower is better. "Current" denotes current visual features. For UniVLA and Moto latent actions, "continuous" denotes encoder outputs before vector quantization; "quantized" denotes the corresponding codebook embeddings.

Probe input	RoboCasa-GR1-Tabletop	RoboTwin Clean + Random	RoboDojo Real + Sim
Training action mean (no visual input)	0.958 ± 0.060	0.996 ± 0.039	1.005 ± 0.014
Current visual features	0.657 ± 0.078	0.587 ± 0.032	0.541 ± 0.005
Visual feature sequence	0.605 ± 0.083	0.477 ± 0.052	0.361 ± 0.013
Current + UniVLA latent actions (quantized)	0.637 ± 0.079	0.526 ± 0.034	0.493 ± 0.006
UniVLA latent actions (quantized)	0.867 ± 0.063	0.787 ± 0.036	0.828 ± 0.012
Current + UniVLA latent actions (continuous)	0.614 ± 0.075	0.432 ± 0.040	0.383 ± 0.004
UniVLA latent actions (continuous)	0.688 ± 0.067	0.492 ± 0.045	0.500 ± 0.006
Current + Moto latent actions (quantized)	0.660 ± 0.076	0.448 ± 0.034	0.387 ± 0.004
Moto latent actions (quantized)	0.817 ± 0.063	0.541 ± 0.038	0.538 ± 0.015
Current + Moto latent actions (continuous)	0.650 ± 0.067	0.418 ± 0.036	0.339 ± 0.005
Moto latent actions (continuous)	0.747 ± 0.057	0.472 ± 0.039	0.423 ± 0.009
Current + WALA latent actions	$\mathbf{0.541 \pm 0.070}$	0.299 ± 0.034	$\mathbf{0.308 \pm 0.006}$
WALA latent actions	0.550 ± 0.061	$\mathbf{0.287 \pm 0.036}$	0.358 ± 0.011

Clean-to-Random transfer. We further evaluate whether an action probe fitted on RoboTwin Clean transfers to Random. Both settings contain the same manipulation tasks. Clean uses plain backgrounds and uncluttered tabletops, whereas Random varies background and tabletop textures, adds randomly placed distractor objects, and randomizes lighting and tabletop height (Chen et al., 2026c). For each task, we use 45 Clean episodes for fitting, 5 Clean episodes for hyperparameter selection, and 50 Random episodes for testing. We retain the action windows and targets described above and repeat the experiment with seeds 42, 43, and 44. PCA, feature scaling, and action normalization are fitted using only the Clean training split. We report the same standardized action MSE as Table 12, using these Clean-training statistics. The visual baselines concatenate current features with either absolute future features or their differences from the current features, $[v_t;\ \{v_{t+k}\}_{k\in\mathcal{K}}]$ and $[v_t;\ \{v_{t+k} - v_t\}_{k\in\mathcal{K}}]$, respectively. We use the same per-group PCA procedure as above. UniVLA and Moto latent actions in this experiment are taken before vector quantization.

Table 13: Action-probe transfer from RoboTwin Clean to Random. Standardized action MSE is reported as mean $\pm$ sample standard deviation over three seeds; lower is better.

Probe input	Standardized action MSE ↓
Absolute visual feature sequence	0.9408 ± 0.0782
Delta visual feature sequence	0.9219 ± 0.0654
UniVLA latent actions	0.7173 ± 0.0249
Moto latent actions	2.7119 ± 0.1852
WALA latent actions	$\mathbf{0.4958 \pm 0.0248}$

Table 13 shows the lowest error with WALA latent actions, outperforming UniVLA (0.7173) and Moto (2.7119). This comparison favors WALA's representation for transferring action decoding across visual settings, beyond fitting actions within a shared setting. Emphasizing action-relevant semantic and geometric changes may help retain information useful across settings while reducing the probe's dependence on setting-specific appearance. Visual deltas also yield a slightly lower mean error than absolute future features.

B.4 Effect of Decoder Supervision and Fine-Tuning

Matching a frozen LAM encoder's outputs transfers pretrained knowledge to the policy. We examine whether additional guidance from the LAM decoder improves future prediction from policy-generated latent actions on downstream interactions. Using RoboDojo videos not seen during LAM pretraining, we compare training without decoder supervision, supervision through a frozen decoder, and joint decoder fine-tuning.

Protocol. We pool real and simulated RoboDojo data from 53 task–embodiment groups, using 100 trajectories per group. For each group, we use 80 trajectories for training and the remaining 20 for evaluation. Training windows start every four frames and contain a 50-step action chunk, with observations at offsets $[0, 12, 25, 38, 50]$. Each setting is trained for one epoch on 16 Alibaba Cloud PPUs with a global batch size of 128. Seeds 42, 43, and 44 require 5,021, 5,030, and 5,025 optimizer updates, respectively. We evaluate the final checkpoint from each run. Within each seed, all three settings share the same data splits and policy initialization. All start from Qwen3-VL-4B-Instruct and the pretrained WALA LAM; the LAM encoder and visual encoders remain frozen. The learning rates are 10^{-5} for the policy backbone and trainable decoder parameters, and 10^{-4} for the action head and adapters.

Compared settings. All three settings retain robot action supervision and latent-action distillation. Setting A omits decoder prediction supervision. Setting B adds the semantic-geometric prediction loss through a frozen decoder, allowing gradients to update the policy and decoder adapter. Setting C also updates the decoder-specific parameters; shared encoder–decoder components remain frozen. For prediction evaluation, A sends policy tokens through its trained distillation adapter to the pretrained decoder, whereas B and C use their trained decoder adapters.

Metrics. We report semantic-feature delta MSE and depth-map delta MSE on real and simulated RoboDojo test trajectories. Table 14 reports the mean and standard deviation across three random seeds.

Table 14: Effect of decoder supervision and fine-tuning on real and simulated RoboDojo data. Semantic and depth delta MSE are reported as mean ± standard deviation across three random seeds after one training epoch. All settings retain robot action supervision and latent-action distillation.

Setting	Semantic delta MSE ↓	Depth delta MSE ↓
A: No decoder supervision	0.03365 ± 0.00101	0.7018 ± 0.0028
B: Frozen decoder	0.02167 ± 0.00020	0.3498 ± 0.0041
C: Trainable decoder	$\mathbf{0.01912 \pm 0.00002}$	$\mathbf{0.3355 \pm 0.0037}$

Results. Adding prediction supervision through a frozen decoder reduces semantic delta MSE by 35.6% and depth delta MSE by 50.2% relative to A. Prediction therefore improves even without updating the decoder, supporting the value of decoder-based prediction supervision in addition to latent-action distillation. Jointly adapting the decoder further reduces the two errors by 11.8% and 4.1%, with a larger relative improvement in semantic prediction. These results support using the pretrained decoder to guide downstream learning and adapting its predictions to the downstream interaction data.

B.5 Action Readout from All Backbone Tokens

Protocol. We examine whether the action head benefits from access to all final-layer VLM tokens rather than only the supervised latent action tokens. The all-token variant uses a single cross-attention resampler with 32 learned queries and 8 attention heads to aggregate the variable-length backbone sequence into 32 features for the original MLP action head. Encoder distillation and decoder prediction still supervise the original latent action tokens. The data, loss functions, and action-chunk length follow the standard RoboCasa configuration; the resampler introduces additional parameters.

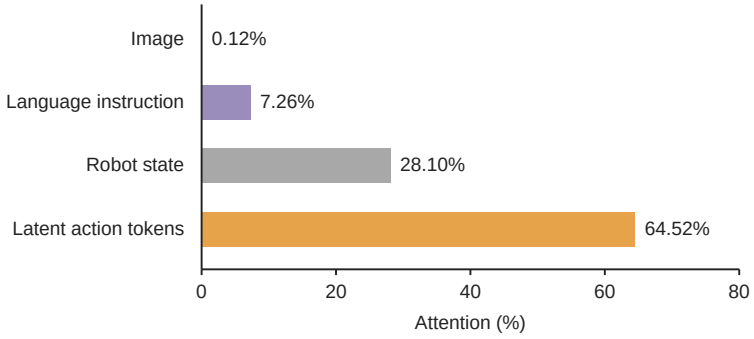

Figure 11: Action-readout attention in the all-token variant on RoboCasa. Statistics cover 100 samples per task across 24 tasks. Latent action tokens receive the highest average attention.

Manipulation performance. The all-token variant achieves 73.3% average success, compared with 75.2% for WALA's latent-token-only action readout, a difference of 1.9 percentage points. Broader access to backbone features therefore does not improve on the reported WALA result. This comparison addresses the design question raised in the Introduction: latent actions can form the direct interface to robot control rather than serve only as an auxiliary prediction target. WALA concentrates interaction-prediction and action-generation supervision on this interface, so the action head reads a representation explicitly shaped by both objectives.

Attention analysis. For each task, we sample 100 distinct training trajectories and one random time step per trajectory, yielding 2,400 observations. The diagnostic inputs include the image, language instruction, and robot state. We average the resampler's attention over its 8 heads and 32 queries, divide each category's attention by its token count, normalize across the four categories, and average over samples; formatting tokens are excluded from the statistics. Figure 11 shows that latent action tokens receive the highest average attention (64.52%), followed by robot state (28.10%), language instructions (7.26%), and image tokens (0.12%). Latent action tokens rank

first in all 24 tasks. Even when the action head can access the full backbone sequence, its readout preferentially attends to the world- and action-supervised latent representation. Together with the success-rate comparison, this pattern supports using latent action tokens as the action head's sole input: access to the broader sequence neither displaces their prominent role nor yields a higher reported success rate. The substantial attention to robot state is consistent with its role in relating the robot's current configuration to subsequent movements. Through contextual processing in the backbone, latent action tokens can also aggregate image, language, and robot-state information into a representation jointly shaped by world and action supervision. Directly decoding actions from these tokens thus provides an efficient and lightweight approach to robot control while maintaining strong manipulation performance.