WHAT SHOULD ROBOT POLICIES LEARN TO PREDICT? RETHINKING SUPERVISION TARGETS IN VISION-LANGUAGE-ACTION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) policies are typically trained with action prediction as the primary supervision signal, since robots ultimately execute actions. Auxiliary objectives such as latent-action (LA) and future-vision (FV) prediction are often added to improve policy learning, yet they compete with action prediction for the same training budget. We ask whether, under a fixed budget, allocating a substantial share of training samples to such objectives, rather than to further action prediction, can improve policy learning. We build a unified VLA framework that couples a two-stream multimodal diffusion transformer (MMDiT) with a sample-level router, which assigns each training sample to either action prediction or an auxiliary objective and thus makes supervision allocation explicit. Within this framework, we instantiate three LA targets based on LAPA and its DINO-feature variants, and three FV targets based on DINOv3, V-JEPA 2, and Wan2.2 representations. Across LIBERO, CALVIN, and four real-robot tasks, suitable LA and FV supervision outperforms action-only training under the same budget, even when action prediction receives no more than half of the training samples. FV generally outperforms LA, although the strongest LA target remains competitive on CALVIN. Diagnostic analyses suggest that DINO-based LA targets emphasize relative motion, whereas FV targets emphasize absolute position and task identity. We therefore argue that what robot policies learn to predict, and how training is allocated among these targets, should be treated as explicit design choices rather than defaulting to action-dominant training.

1 INTRODUCTION

Vision-language-action (VLA) models have emerged as a prominent paradigm for robot learning, achieving strong performance across many manipulation tasks (Kim et al., 2025; Black et al., 2025a). Beyond direct action supervision, recent studies increasingly introduce additional learning objectives into VLA training, including geometric representation alignment and spatial or 3D structure prediction (Li et al., 2026; Guo et al., 2025). Among these, latent-action (LA) and future-vision (FV) prediction have become two representative forms of supervision. Latent action abstracts visual transitions into compact tokens that characterize how the scene changes over time (Ye et al., 2025; Chen et al., 2026), whereas future vision encourages the policy to predict future representations, providing supervision about how the scene is expected to evolve (Wu et al., 2024; Zheng et al., 2025).

Because executable actions are the ultimate outputs of robot policies, direct action prediction is typically treated as the dominant training objective, while LA and FV prediction are introduced as auxiliary supervision. Prior studies have examined latent-action formulations and integration strategies (Lin et al., 2026; Bu et al., 2026), showing that such supervision can benefit downstream policy learning. More importantly, LA and FV are not merely auxiliary in effect: by providing learning signals about state changes and task structure, they directly shape the same policy during optimization. Yet, showing that an additional target is useful does not establish that direct action prediction is itself the most effective supervision. We therefore ask (Figure 1): *under a fixed training budget, what should robot policies learn to predict, and can assigning LA or FV prediction an equal or larger share of training samples than direct action prediction improve action execution?*

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.

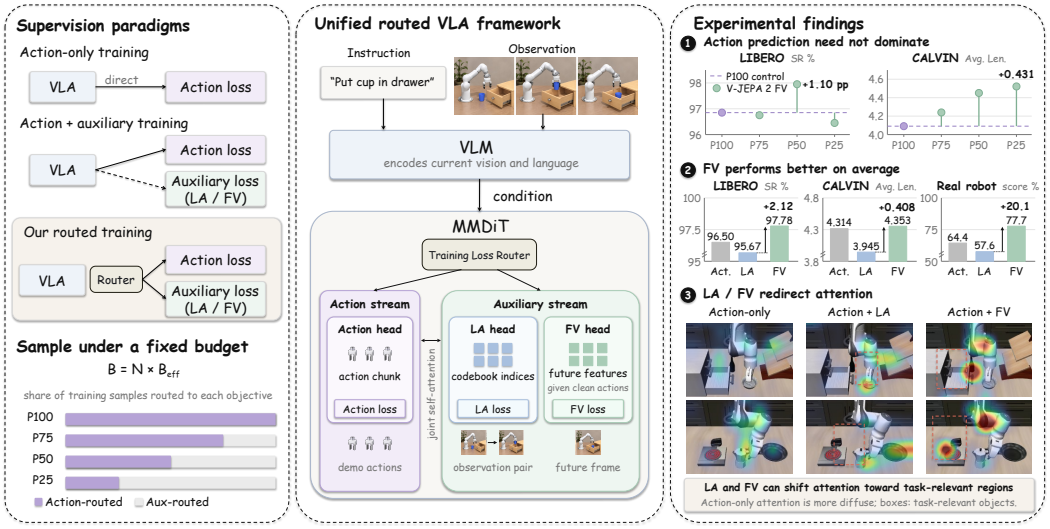

Figure 1: **Overview. Left:** under a fixed training budget, a router sends each sample to the action loss or one auxiliary loss (LA or FV); Px routes x% of samples to the action loss. **Middle:** two-stream VLA; FV prediction conditions on clean demonstrated actions. **Right:** (1) V-JEPA 2 FV vs. its source-matched P100 control; (2) LA and FV means over three targets each at P50 vs. action-only (Act.), arrows marking unrounded FV−LA (real robot: % of max score); (3) per-map normalized LIBERO attention for action-only, +LA (LAPA-DINOv3), and +FV (V-JEPA 2).

To investigate this question, we build a unified VLA framework that places a two-stream MMDiT on top of a pretrained VLM, with separate action and auxiliary experts interacting through joint self-attention. We further introduce a sample-level router that assigns each training sample to either action or auxiliary supervision under a fixed training budget, making their relative training allocation explicit. Within this framework, we instantiate three LA targets and three FV targets: LA predicts discrete latent tokens extracted by different encoders, whereas FV predicts future features in different visual representation spaces. This unified setup enables a controlled comparison of how different supervision targets shape policy learning, with source-matched controls that change only routing for each FV target; FV prediction still conditions on demonstrated actions.

We evaluate our framework on two simulation benchmarks (LIBERO and CALVIN) and four real-robot manipulation tasks, and obtain three main findings. First, **suitable LA and FV supervision can outperform action-only training** under the same training budget, even when direct action prediction receives only half or fewer of the training samples. Second, **FV generally performs better than LA** across both simulation and real-robot tasks, while the strongest LA configuration remains competitive on CALVIN. Third, our analyses suggest that **LA and FV shape policy learning in different ways**: DINO-based LA targets emphasize relative motion, whereas FV targets better capture absolute position and task identity, and both can alter policy attention toward task-relevant regions.

Our contributions are threefold: **(1)** We revisit the dominant role of direct action prediction in VLA training and investigate how allocating supervision to LA and FV affects policy learning. **(2)** We introduce a unified VLA training framework with a sample-level training router that explicitly controls the allocation of action and auxiliary supervision under a fixed training budget, enabling controlled comparisons across multiple LA and FV targets. **(3)** We conduct extensive experiments on LIBERO, CALVIN, and four real-robot manipulation tasks, together with representation and attention analyses, and derive systematic findings on action-supervision allocation, the performance differences between LA and FV, and their representation characteristics.

2 Related Work

Supervision beyond action labels. Robot learning draws on supervision beyond action labels, including visual reconstruction and language generation (Karamcheti et al., 2023), vision-language

co-training (Zitkovich et al., 2023), and geometric alignment (Li et al., 2026; Guo et al., 2025; Yang et al., 2026). We focus on two demonstration-derived targets, LA and FV, and study both which target to predict and how to allocate training samples between it and direct action prediction.

Latent actions. LAPA and UniVLA pretrain policies with latent actions derived from visual transitions (Ye et al., 2025; Bu et al., 2025). The villa-X framework jointly models latent and robot actions, using latent plans to support action generation (Chen et al., 2026). Analyses examine controllable changes and exogenous noise (Zhang et al., 2025), while LARYBench (Nie et al., 2026) finds that general visual encoders can outperform specialized latent-action models on its semantic and physical-action probes. Empirical studies compare latent-action formulations, integration strategies, and loss weights (Lin et al., 2026), and examine links between representation-quality proxies and downstream policy performance (Bu et al., 2026). Rather than reweighting losses on every sample, we route each sample to a single objective, and we probe LA and FV sources with the same absolute- and relative-motion readouts.

Future vision. GR-1, GR-2, UWM, and DreamZero jointly model future images or videos and robot actions (Wu et al., 2024; Cheang et al., 2024; Zhu et al., 2025; Ye et al., 2026). Seer and VPP guide action generation with predicted visual states or features (Tian et al., 2025; Hu et al., 2025). FLARE enhances policy learning through future-representation alignment and compares against an action-only baseline trained for longer (Zheng et al., 2025). We instead compare against action-only training with the same total budget, vary the action share, and add source-matched FV controls, including allocations where auxiliary prediction receives most samples.

3 METHOD

3.1 PROBLEM SETUP AND SUPERVISION TARGETS

Problem setup. At time t, the policy receives the current visual observation o_t, the language instruction ℓ, and the robot state s_t, and predicts an action chunk over the next H steps:

$$\mathbf{a}_t = (a_t, \dots, a_{t+H-1}), \qquad \hat{\mathbf{a}}_t \sim \pi_\theta(\cdot \mid o_t, \ell, s_t). \tag{1}$$

We study the allocation of supervision under a fixed training budget. Suppose training runs for N optimizer updates with effective batch size B_{eff}. For a selected auxiliary prediction target $q \in \{\text{LA}, \text{FV}\}$, we define the total training budget and its allocation between action and auxiliary prediction as $\mathcal{B} = NB_{\text{eff}} = \mathcal{B}_A + \mathcal{B}_q$. Here $\mathcal{B}$ counts training-sample occurrences, including repeated uses of the same demonstration. Matching this budget fixes sample exposure; it does not establish equal FLOPs or wall-clock time.

Latent action (LA). We encode each visual transition into a group of discrete LA tokens using an LA encoder $E^{(m)}_{\text{LA}}$:

$$c_u = E^{(m)}_{\text{LA}}(o_u, o_{u+\delta}), \qquad m \in \{\text{LAPA}, \text{LAPA-DINOv2}, \text{LAPA-DINOv3}\}. \tag{2}$$

Here δ is the temporal offset between the observations used to construct a transition label, and $c_u = (c_{u,1}, \dots, c_{u,K})$ contains K discrete codebook indices. The policy predicts the concatenation $z_t = (c_t, \dots, c_{t+H-1})$, flattened into $L_z = HK$ token positions, so that the target spans the same H steps as the action chunk. We use $H = 8$; the benchmark-specific temporal construction is given in Appendix A.1. LAPA encodes raw-image changes, whereas LAPA-DINOv2 and LAPA-DINOv3 (collectively, LAPA-DINO) encode changes between DINOv2 and DINOv3 visual representations. The same appendix gives token counts and codebooks.

Future vision (FV). We use an FV encoder $E^{(m)}_{\text{FV}}$ to encode the current and future visual observations:

$$f_t = E^{(m)}_{\text{FV}}(o_t), \qquad f_{t+\delta} = E^{(m)}_{\text{FV}}(o_{t+\delta}), \qquad m \in \{\text{DINOv3}, \text{V-JEPA 2}, \text{Wan2.2 VAE}\}. \tag{3}$$

Both f_t and $f_{t+\delta}$ are continuous visual representations. We use f_t as an MMDiT input and construct the FV flow-matching target from $f_{t+\delta}$, with $\delta = 8$ for FV. Detailed input formats and feature dimensions for each encoder are provided in Appendix A.1. For both LA and FV, we refer to the encoder m as the *source* of the target.

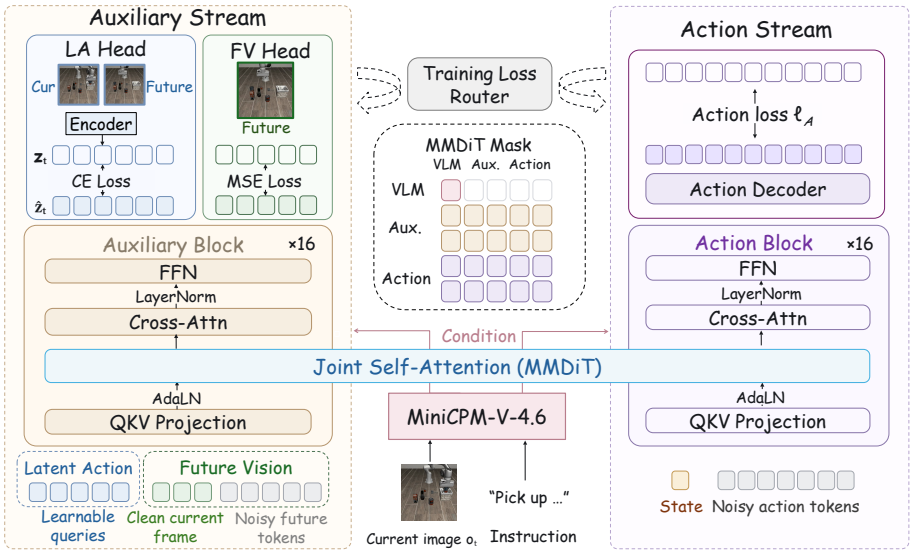

Figure 2: VLM–MMDiT architecture. Action and auxiliary (LA/FV) experts interact through joint self-attention; sample-level routing selects the active loss. Future observations are used only for auxiliary training.

3.2 Policy Architecture and Routing

VLM conditioning. We use a pretrained VLM to encode the current visual observation and language instruction as $h_t = \mathrm{VLM}_\phi(o_t, \ell)$, where h_t denotes the last hidden states of the VLM. We use MiniCPM-V-4.6 and condition both MMDiT experts on h_t through cross-attention, while s_t separately conditions the action expert.

MMDiT prediction head. The VLM is followed by a two-stream MMDiT prediction head (Figure 2) consisting of an action expert and an auxiliary expert. The two experts use separate input projections and feed-forward networks and exchange information through joint self-attention.

Sample-level training router. To allocate supervision across predictive targets under a fixed training budget, we route each training sample to either the action prediction objective or the selected LA or FV prediction objective. The router determines the active loss and the MMDiT input construction, which differ between routes (e.g., FV-routed samples condition on clean demonstrated actions, whereas action-routed samples receive noisy actions); each sample therefore contributes a single loss rather than a weighted sum of both, and each sample occurrence counts toward exactly one of $\mathcal{B}_A$ and $\mathcal{B}_q$. Both experts participate in joint self-attention on either route. For a selected auxiliary target $q \in \{\mathrm{LA}, \mathrm{FV}\}$, we independently sample a routing indicator r_i for each training example. The routing distribution and expected allocations under the fixed budget $\mathcal{B}$ are

$$r_i \sim \mathrm{Bernoulli}(p_A), \qquad \mathbb{E}[\mathcal{B}_A] = p_A \mathcal{B}, \qquad \mathbb{E}[\mathcal{B}_q] = (1 - p_A)\mathcal{B}. \tag{4}$$

Here $r_i = 1$ denotes the action route and $r_i = 0$ denotes the auxiliary route. We denote $p_A = 0.25,\ 0.50,\ 0.75$, and 1.00 as P25, P50, P75, and P100, respectively. These proportions count only samples assigned to the direct action loss, not all uses of action labels (FV-routed samples also condition on demonstrated actions) or relative gradient magnitudes. For each FV configuration, the *source-matched control* is its P100 version: it keeps the same FV encoder, MMDiT input f_t, two-stream architecture, and training settings, but every sample follows the action route, so the FV loss is never active.

Auxiliary expert configurations. The auxiliary expert is instantiated differently for LA and FV supervision. For LA, we introduce L_z learnable query tokens Q_{LA}, each corresponding to one LA token position. The LA expert predicts

$$\hat{z}_t \sim \pi_\theta^{\mathrm{LA}}(\cdot \mid Q_{\mathrm{LA}}, h_t), \tag{5}$$

where z_t is the concatenated LA target defined in Section 3.1. Since Q_{LA} carries no source-specific features, LA configurations use the shared full-budget action-only baseline as their P100 reference. For FV, the auxiliary expert takes the current visual representation f_t together with a noisy future visual representation $f_{t+\delta}^{\tau_f}$ and the action $\mathbf{a}_t$, and predicts the flow velocity of the future visual representation:

$$\hat{v}_{t+\delta}^{f} = v_\theta^{\mathrm{FV}}\left(f_t, f_{t+\delta}^{\tau_f}, \mathbf{a}_t, h_t, \tau_f\right), \tag{6}$$

where $\tau_f \in [0,1]$ denotes the flow time. On FV-routed training samples, $\mathbf{a}_t$ is the clean demonstrated action chunk. The FV expert is trained with flow matching (Section 3.3).

Action expert. The action expert takes the robot state s_t and a noisy action chunk $\mathbf{a}_t^{\tau_a}$, and predicts the clean action chunk conditioned on the VLM hidden states h_t:

$$\hat{\mathbf{a}}_t \sim \pi_\theta^A\left(\cdot \mid \mathbf{a}_t^{\tau_a}, s_t, h_t, \tau_a\right), \tag{7}$$

where $\tau_a \in [0,1]$ is the action noise time and $\mathbf{a}_t^{\tau_a}$ is the corresponding noisy action input. At inference, we use the same action-route computation as for action-routed training samples. Both MMDiT experts participate in joint self-attention, but no future observation or auxiliary target is provided, and only the action-expert output is decoded and executed.

3.3 Training Objectives

Noise construction. For action and FV prediction, we sample independent Gaussian noise $\epsilon_a, \epsilon_f \sim \mathcal{N}(0, I)$. With $\tau_a, \tau_f \in [0,1]$, the noisy inputs and FV flow velocity are

$$\begin{aligned} \mathbf{a}_t^{\tau_a} &= (1-\tau_a)\epsilon_a + \tau_a \mathbf{a}_t, \\ f_{t+\delta}^{\tau_f} &= (1-\tau_f)\epsilon_f + \tau_f f_{t+\delta}, \qquad v_{t+\delta}^{f} = f_{t+\delta} - \epsilon_f. \end{aligned} \tag{8}$$

Prediction losses. The action objective and auxiliary objectives are

$$\mathcal{L}_A = \ell_A(\hat{\mathbf{a}}_t, \mathbf{a}_t), \qquad \mathcal{L}_{\mathrm{LA}} = \mathrm{CE}(\hat{z}_t, z_t), \qquad \mathcal{L}_{\mathrm{FV}} = \left\| \hat{v}_{t+\delta}^{f} - v_{t+\delta}^{f} \right\|_2^2. \tag{9}$$

Here ℓ_A is the action loss defined by each training dataset, whose clean-action parameterization and reduction are specified in Appendix A.2; $\mathcal{L}_{\mathrm{LA}}$ is averaged over valid LA tokens.

Routed objective. Each sample contributes only the loss associated with its selected route. For a batch of B samples, the overall training objective is

$$\mathcal{L} = \frac{1}{B}\sum_{i=1}^{B}\left[r_i \mathcal{L}_{A,i} + (1-r_i)\lambda_q \mathcal{L}_{q,i}\right], \qquad q \in \{\mathrm{LA}, \mathrm{FV}\}, \tag{10}$$

where λ_q is a target-specific loss weight, tuned for LA and fixed across allocations (Appendix B.4).

4 Experiments

We investigate the roles of LA and FV in policy learning through simulation benchmarks, real-robot experiments, and representation analyses, focusing on three research questions:

Question 1. Must direct action prediction receive most supervision?

Question 2. How do LA and FV compare across training regimes and tasks?

Question 3. Why can LA and FV supervision help policy learning?

4.1 Experimental Setup

Benchmarks. We evaluate on LIBERO (Liu et al., 2023) across its Spatial, Object, Goal, and Long suites using task success rates. On CALVIN (Mees et al., 2022), we follow ABC→D and report average task-chain length (Avg. Len.) and five-task success (SR-5). Evaluation protocols are detailed in Appendix A.3. We also evaluate four real-robot tasks involving basket pick-and-place and cup/object rearrangement, scored by completion progress.

Table 1: LIBERO and CALVIN policy performance. Mixed configurations use P50 and the full budget. Parentheses: differences from the half-budget baseline. S/O/G/L: Spatial/Object/Goal/Long; "–": unavailable. Source-matched FV controls: Appendices B.2, B.5, and B.6.

	Method	Param. (B)	LIBERO					CALVIN	
			S	O	G	L	Avg. ↑	SR-5 ↑	Avg. Len. ↑
Ref.	OpenVLA	≈8	84.7	88.4	79.2	53.7	76.5	43.5	3.270
	π_0	≈3	96.8	98.8	95.8	85.2	94.2	60.1	3.840
	UniVLA	≈7	96.5	96.8	95.6	92.0	95.2	56.5	3.800
	$\pi_{0.5}$	≈3	98.8	98.2	98.0	92.4	96.9	–	–
Act.	Half budget	≈2	97.2	99.4	94.6	81.6	93.20	57.7	3.971
	Full budget	≈2	98.2	98.6	98.2	91.0	96.50 (+3.30)	70.3 (+12.6)	4.314 (+0.343)
LA	LAPA	≈2	96.4	98.2	94.2	89.8	94.65 (+1.45)	42.1 (−15.6)	3.466 (−0.505)
	LAPA-DINOv2	≈2	97.0	99.6	98.0	91.2	96.45 (+3.25)	63.4 (+5.7)	3.935 (−0.036)
	LAPA-DINOv3	≈2	97.2	97.2	96.8	92.4	95.90 (+2.70)	**77.0** (+19.3)	4.434 (+0.463)
FV	DINOv3	≈2	98.4	99.2	98.4	93.0	97.25 (+4.05)	75.1 (+17.4)	4.370 (+0.399)
	V-JEPA 2	≈2	98.8	99.2	98.6	95.2	97.95 (+4.75)	76.0 (+18.3)	**4.451** (+0.480)
	Wan2.2	≈2	98.4	100.0	99.0	95.2	**98.15** (+4.95)	70.5 (+12.8)	4.237 (+0.266)

Training configurations. All policies use MiniCPM-V-4.6 in the unified framework (Section 3). On LIBERO and CALVIN, the *full budget* is 60k optimizer updates and the *half budget* is 30k; results use the full budget unless noted. The full-budget action-only baseline spends the entire budget on action prediction, and the half-budget baseline matches the expected number of action-routed samples in a full-budget P50 run. Detailed configurations and evaluation protocols are provided in Appendices A and B.8.

4.2 Must Direct Action Prediction Receive Most Supervision?

We first evaluate an equal allocation between action and auxiliary prediction (P50; Table 1). Against the half-budget action-only baseline, which receives the same expected number of action samples, all six configurations improve LIBERO success, and all FV configurations and LAPA-DINOv3 improve CALVIN average task-chain length, whereas LAPA and LAPA-DINOv2 do not. Against the full-budget baseline, which spends the entire budget on action prediction, all three FV configurations exceed its 96.50% on LIBERO (up to 98.15% with Wan2.2), and DINOv3, V-JEPA 2, and LAPA-DINOv3 exceed its 4.314 on CALVIN (up to 4.451 with V-JEPA 2). Suitable configurations can thus outperform action-only training while assigning only half of their samples to direct action prediction, although no LA configuration does so on LIBERO. Since configurations also differ in visual inputs and auxiliary experts, we next examine allocation with the corresponding controls.

Figure 3 varies the action share at a fixed total budget. Each FV configuration uses a source-matched P100 control that differs only in routing; LA uses the shared full-budget action-only baseline. On CALVIN, V-JEPA 2 at P25 reaches an average task-chain length of 4.522 versus 4.091 for its source-matched control, also exceeding the full-budget baseline in Table 1. This configuration assigns 75% of training samples to FV prediction in expectation, showing that direct action prediction need not receive most samples to improve task performance.

This does not imply that reducing action prediction always helps. On LIBERO, V-JEPA 2 and Wan2.2 outperform their source-matched controls at P50, but every tested P25 configuration falls below its P100 control. The appropriate allocation therefore depends on both the target and the benchmark.

Figure 4 fixes P50 to examine whether the gains persist across training budgets. On CALVIN, all three FV configurations exceed their source-matched controls in average task-chain length at all four tested budgets. These controls, however, score below the shared action-only baseline at every budget, so gains over that baseline are less consistent: V-JEPA 2 exceeds it at three of the four budgets, and among LA configurations only LAPA-DINOv3 does so, at the full budget. On LIBERO (Figure 8 in Appendix B.3), gains vary with the target and budget and do not occur at every stage. The joint allocation–budget sweep in Appendix B.7 (Figure 9) further shows that CALVIN gains

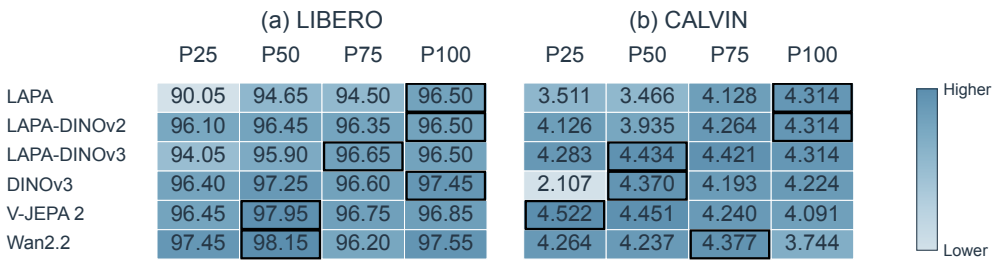

Figure 3: Allocation at the full budget. Cells show LIBERO success (%) and CALVIN Avg. Len.; darker is higher within each benchmark. Black outlines mark each model's best allocation.

over source-matched controls for V-JEPA 2 and Wan2.2 extend across all tested mixed allocations and budgets.

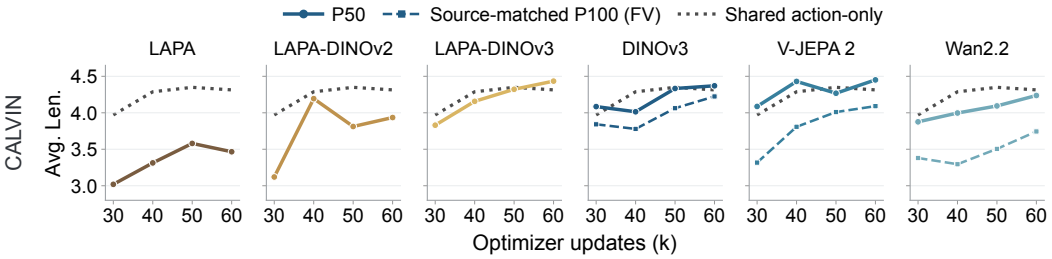

Figure 4: CALVIN performance across budgets at P50. Solid: P50; colored dashed: source-matched FV P100; gray dotted: shared action-only baseline, also the LA P100 reference. LIBERO: Figure 8.

4.3 How Do LA and FV Compare Across Training Regimes and Tasks?

We next compare the complete LA and FV configurations, which differ in target sources and auxiliary experts but share the action objective.

On LIBERO, FV leads overall. At P50 and the full budget (Table 1), all three FV configurations outperform all three LA configurations in average success and on Spatial, Goal, and Long, while scores overlap on Object. The highest LIBERO score at each tested mixed allocation also comes from FV (Figure 3), and FV generally scores higher than LA across the four P50 budgets (Figure 8). On CALVIN, the comparison depends on allocation and budget. At P50 with the full budget, LAPA-DINOv3 nearly matches the best FV configuration, V-JEPA 2 (4.434 vs. 4.451), and their order reverses on SR-5 (77.0% vs. 76.0%). In average task-chain length, V-JEPA 2 leads at P25 and P50, whereas LAPA-DINOv3 leads at P75 (4.421). Across P50 budgets (Figure 4), all FV configurations have longer average task-chain lengths than all LA configurations at the half budget, but LAPA-DINOv3 becomes competitive with the best FV configurations at 50k updates and the full budget.

On the four real-robot tasks, all configurations use the same training budget, with LA and FV evaluated at P50 (Appendix B.8). Figure 5 reports partial-credit task-completion scores over 20 trials per task. Every FV configuration scores higher than every LA configuration and the action-only baseline on all four tasks, with Wan2.2 highest throughout, whereas no LA configuration exceeds the action-only baseline on any task. Overall, FV thus generally outperforms LA and, on the real robot, again surpasses action-only training with half of its samples routed to action prediction.

4.4 Why Can LA and FV Supervision Help Policy Learning?

To investigate possible explanations for these gains, we examine action-related information recoverable from frozen source representations, its organization in feature space, and visual attention in trained policies.

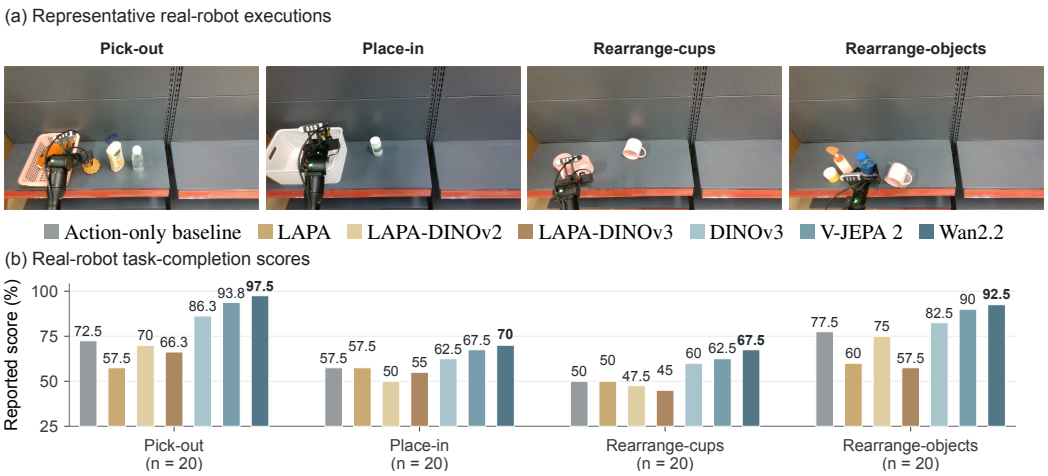

Figure 5: Real-robot evaluation: (a) representative executions; (b) partial-credit scores normalized by 160 points per task (20 trials, up to 8 points each). LA/FV use P50. Protocol: Appendix B.8.

We begin by testing what action information can be recovered from the six frozen source representations. Separate visual-only probes predict absolute action trajectories (ABS) or relative action changes (Δ_c) over a 45-step segment from the frozen representations of its image pair (for LA, the post-quantization LA vector), on CALVIN environment-D validation and on unseen tasks of the dual-arm real-robot dataset AgiBotWorld-Beta (AgiBot-World-Contributors et al., 2025), using LARYBench splits (Nie et al., 2026) (Appendix C). The readouts in Table 2 indicate that both LA and FV carry action-related information, so predicting such targets may provide indirect action supervision on auxiliary-routed samples. Their strengths differ: all three FV sources yield lower ABS position errors than all three LA sources on both datasets, whereas both LAPA-DINO variants yield lower relative-motion errors than all three FV sources at every Δ_c setting on unseen AgiBotWorld-Beta tasks; on CALVIN, the leading encoder changes with c. Plain LAPA, which yields the weakest CALVIN readouts, is also the weakest policy configuration on CALVIN (Table 1).

Table 2: Frozen-source position errors (mm; lower is better) from visual-only probes; Δ_c uses action differences at stride c within a 45-step segment. Bold: column minima. Protocol: Appendix C.1.

	AgiBotWorld-Beta unseen tasks				CALVIN environment-D validation			
Encoder	ABS	$\Delta_{c=1}$	$\Delta_{c=9}$	$\Delta_{c=45}$	ABS	$\Delta_{c=1}$	$\Delta_{c=9}$	$\Delta_{c=45}$
DINOv3	**76.31**	25.23	13.73	14.19	**52.06**	69.64	75.44	58.21
V-JEPA 2	101.98	20.44	11.42	12.16	54.70	64.21	68.43	55.85
Wan2.2	105.55	32.28	17.20	16.65	53.86	53.44	59.72	**50.01**
LAPA-DINOv2	155.65	14.79	**8.62**	**8.79**	62.70	56.89	59.16	54.40
LAPA-DINOv3	155.61	**14.68**	8.71	9.23	60.14	**53.23**	**57.41**	50.48
LAPA	164.82	30.79	15.79	15.58	127.91	139.59	109.36	95.03

We next examine how this action-related information is organized in feature space. On AgiBotWorld-Beta seen validation (held-out segments of the probe-training tasks), we correlate representation distances between sample pairs with distances between their net action changes over the segment. Figure 6a plots this correlation against the normalized mean squared error (NMSE) of a second relative-motion probe (delta-step5), which predicts the nine successive five-step action differences within the segment. Both LAPA-DINO variants combine higher correlations with lower readout errors than the FV sources; LAPA-DINOv3 has the highest correlation (0.375) and lowest NMSE (0.807). This alignment is consistent with LAPA-DINO targets providing a learning signal that distinguishes different movements.

Clustering on the same validation split further distinguishes motion-related and task-related organization (Figure 6b). We measure normalized mutual information (NMI) between feature-cluster assignments and motion labels obtained by clustering ground-truth displacements, or dataset task

IDs. Both LAPA-DINO variants associate more strongly with motion labels than any FV source, whereas all three FV sources associate more strongly with task IDs than any LA source. Together with the readouts, these results suggest that LAPA-DINO supervision may emphasize motion distinctions, whereas FV supervision may supply position- and task-associated cues. Different tasks may benefit differently from these cues, potentially contributing to the setting-dependent comparisons in Section 4.3.

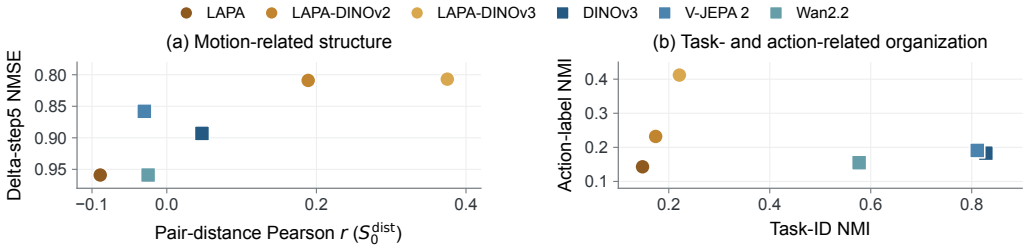

Figure 6: Frozen-source diagnostics on AgiBotWorld-Beta seen validation. (a) Correlation between net-action-change and representation pair distances, plotted against delta-step5 NMSE. (b) Feature-cluster NMI with displacement-derived action labels and task IDs. Each point is one encoder; details in Appendix C.2.

Finally, Figure 7 compares the action-only policy and all six LA/FV configurations on a LIBERO observation (extraction protocol and a second observation in Appendix C.3). Whereas the action-only map is diffuse, V-JEPA 2 FV highlights the stove region and the robot, and LAPA-DINOv3 emphasizes areas around the gripper and nearby containers. These maps suggest that LA and FV supervision can reshape the policy's attention toward task-relevant objects and interaction regions, which may supply more task-relevant visual information for action prediction and contribute to the gains observed above.



Figure 7: Qualitative policy attention on a LIBERO observation; a second task is shown in Figure 10. Colors show relative intensity within each map.

5 Conclusion

Robot policies must execute actions, but this does not require most training samples to supervise direct action prediction. In a unified VLA framework with sample-level routing, suitable LA and FV targets can outperform action-only training under the same budget even when direct action prediction receives no more than half of the samples, although the best allocation depends on the target, benchmark, and budget. Between the two families, FV generally performs better, leading on LIBERO and all four real-robot tasks, while LAPA-DINOv3 remains competitive on CALVIN. Diagnostics suggest why both can help: both carry action-related information, with LAPA-DINO targets emphasizing relative motion and FV targets emphasizing absolute position and task identity, and both can redirect policy attention toward task-relevant regions. These findings support designing predictive targets and their supervision allocation jointly, rather than defaulting to action-only training.

AI USE STATEMENT

Generative AI tools were used to provide coding suggestions; the research code was written by the authors. AI tools also assisted with the presentation of the manuscript: they were used to draft and edit parts of the text for clarity and consistency, and to help with LaTeX formatting and preparation of the disclosure statements. All AI-assisted text was reviewed and revised by the authors. AI tools were not used to develop research ideas or analyze experimental results. Scientific judgments and final decisions about the manuscript remain with the authors, who take responsibility for the final text, claims, code, and other artifacts.

ETHICS STATEMENT

This study evaluates supervision targets for robot manipulation on LIBERO and CALVIN, with frozen-representation analyses and four real-robot manipulation tasks using a DOS W1 left follower arm. Appendix B.8 describes the real-robot tasks and evaluation protocol. No scene damage occurred in the reported trials. These observations are limited to the tested platform, tasks, and conditions; they do not establish safety for other environments or interactions with people. Deployment beyond these experiments would require a separate assessment of physical risks and appropriate safeguards.

REPRODUCIBILITY STATEMENT

Section 3 specifies the policy architecture, prediction targets, and training objectives. Section 4.1 and Appendix A document the model and target-encoder configurations, label construction, training data, optimization schedules, supervision allocation, control matching, and benchmark evaluation metrics and seeds. Appendix B provides checkpoint and allocation results, real-robot task protocols, and component-level scoring. Appendix C describes the diagnostic datasets and splits, target preprocessing, probe training, metrics, and clustering procedures.

REFERENCES

AgiBot-World-Contributors, Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, Shu Jiang, Yuxin Jiang, Cheng Jing, Hongyang Li, Jialu Li, Chiming Liu, Yi Liu, Yuxiang Lu, Jianlan Luo, Ping Luo, Yao Mu, Yuehan Niu, Yixuan Pan, Jiangmiao Pang, Yu Qiao, Guanghui Ren, Cheng Ruan, Jiaqi Shan, Yongjian Shen, Chengshi Shi, Mingkang Shi, Modi Shi, Chonghao Sima, Jianheng Song, Huijie Wang, Wenhao Wang, Dafeng Wei, Chengen Xie, Guo Xu, Junchi Yan, Cunbiao Yang, Lei Yang, Shukai Yang, Maoqing Yao, Jia Zeng, Chi Zhang, Qinglin Zhang, Bin Zhao, Chengyue Zhao, Jiaqi Zhao, and Jianchao Zhu. AgiBot World Colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025. URL https://arxiv.org/abs/2503.06669.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

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 Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 27–30 Sep 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.

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 2025. doi: 10.15607/RSS.2025.XXI.014.

Xizhou Bu, Qingda Hu, Lei Zhou, Lingfeng Zhang, Yingbo Tang, Zihao Liu, Xinyi Tao, Zhiqiang Ma, Qingqiu Huang, Chufeng Tang, Hongbo Wang, Jing Zhang, Jiayi Ma, Hangjun Ye, Wei Li, and Xiaoshuai Hao. What matters for latent actions in robot learning. *arXiv preprint arXiv:2608.19613*, 2026. URL https://arxiv.org/abs/2608.19613.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, Hanbo Zhang, and Minzhao Zhu. GR-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024. URL https://arxiv.org/abs/2410.06158.

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*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/73471f46f69a13cede6cf2db3a6f438f-Abstract-Conference.html.

Minghao Guo, Meng Cao, Jiachen Tao, Rongtao Xu, Yan Yan, Xiaodan Liang, Ivan Laptev, and Xiaojun Chang. GLaD: Geometric latent distillation for vision-language-action models. *arXiv preprint arXiv:2512.09619*, 2025. URL https://arxiv.org/abs/2512.09619.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 24328–24346. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/hu25g.html.

Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.032.

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 Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial Forcing: Implicit spatial representation alignment for vision-language-action model. In *International Conference on Learning Representations*, pp. 132324–132345, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/d676c05a8fb93ad9231508d95567c498-Abstract-Conference.html.

Yihan Lin, Haoyang Li, Yang Li, Haitao Shen, Yihan Zhao, Chao Shao, and Jing Zhang. From pixels to tokens: A systematic study of latent action supervision for vision-language-action models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=VMsumctGvg.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8c3c666820ea055a77726d66fc7d447f-Abstract-Datasets_and_Benchmarks.html.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022. doi: 10.1109/LRA.2022.3180108.

Meituan LongCat. LARYBench official repository. GitHub repository, 2026. URL https://github.com/meituan-longcat/LARYBench.

Dujun Nie, Fengjiao Chen, Qi Lv, Jun Kuang, Xiaoyu Li, Xuezhi Cao, and Xunliang Cai. LARY: A latent action representation yielding benchmark for generalizable vision-to-action alignment. *arXiv preprint arXiv:2604.11689*, 2026. URL https://arxiv.org/abs/2604.11689.

OpenBMB. MiniCPM-V-4.6 model card. Hugging Face model repository, 2026. URL https://huggingface.co/openbmb/MiniCPM-V-4.6.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seung Eun Yi, Michael 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, Herve Jegou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=2NlGyqNjns. Featured Certification.

Yang Tian, Sizhe Yang, Jia Zeng, Ping Wang, Dahua Lin, Hao Dong, and Jiangmiao Pang. Predictive inverse dynamics models are scalable learners for robotic manipulation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/e5b5c402bb7bd5e60bede6961d6fe39e-Abstract-Conference.html.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. URL https://arxiv.org/abs/2503.20314.

Wan Team. Wan2.2 official repository. GitHub repository, 2025. URL https://github.com/Wan-Video/Wan2.2.

Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/2c37c5bcef24b9541550261dcd63261b-Abstract-Conference.html.

Lishan Yang, Wenxuan Song, Xi Wang, Pingyue Sheng, Zheng Fang, Ziyang Zhou, Junjie He, Haodong Yan, Jiayi Chen, Nan Sun, Qiao Sun, Pengwei Wang, Lingqiao Liu, Yan Wang, Yuxiang Gao, Feras Dayoub, and Haoang Li. 4D-WAM: Infusing spatiotemporal awareness into world action models through trajectory fields. *arXiv preprint arXiv:2608.08023*, 2026. URL https://arxiv.org/abs/2608.08023.

Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Chi Chen, Haoyu Li, Weilin Zhao, Zhihui He, Qianyu Chen, Ronghua Zhou, Zhensheng Zou, Haoye Zhang, Shengding Hu, Zhi Zheng, Jie Zhou, Jie Cai, Xu Han, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. Efficient GPT-4V level multimodal large language model for deployment on edge devices. *Nature Communications*, 16:5509, 2025. doi: 10.1038/s41467-025-61040-5. URL https://www.nature.com/articles/s41467-025-61040-5.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Se June 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*, 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. *arXiv preprint arXiv:2602.15922*, 2026. URL https://arxiv.org/abs/2602.15922.

Chuheng Zhang, Tim Pearce, Pushi Zhang, Kaixin Wang, Xiaoyu Chen, Wei Shen, Li Zhao, and Jiang Bian. What do latent action models actually learn? In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/d7b3cef7c31b94a4a533db83d01a8882-Abstract-Conference.html.

Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loïc Magne, Avnish Narayan, You Liang Tan, Guanzhi Wang, Qi Wang, Jiannan Xiang, Yinzhen Xu, Seonghyeon Ye, Jan Kautz, Furong Huang, Yuke Zhu, and Linxi Fan. FLARE: Robot learning with implicit world modeling. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 3952–3971. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/zheng25a.html.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.015.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

A Training and Evaluation Details

A.1 Target Encoders and Label Construction

Table 3 records the policy VLM and the frozen target encoders used in the reported experiments. The frozen-source diagnostics in Section 4.4 use the same six target encoders and checkpoints.

Table 3: Backbones, checkpoints, and input configurations used in the reported experiments. Parameter counts describe the loaded model or encoder. LA predicts stored integer codes; FV predicts a velocity target constructed from continuous encoder outputs.

Component	Provider and checkpoint	Configuration used
Policy VLM	OpenBMB; `openbmb/MiniCPM-V-4.6` (Yao et al., 2025; OpenBMB, 2026)	Qwen3.5-0.8B language model and SigLIP2-400M vision encoder; approximately 1.30B locally loaded parameters. The text tower has hidden size 1024 and 24 layers; the vision tower has hidden size 1152 and 27 layers. The processor uses a base resolution of 448 and supports dynamic slicing.
LAPA LA	Released LARYBench asset `laq_openx.pt` (Nie et al., 2026; Meituan LongCat, 2026)	Image-based latent-action quantizer; four codes per transition and codebook size eight.
LAPA-DINOv2 LA	Meta FAIR; `facebook/dinov2-large` (Oquab et al., 2024) plus `laq_dinov2.pt` (Nie et al., 2026; Meituan LongCat, 2026)	ViT-L/14, approximately 304M parameters; 24 layers, 16 heads, hidden size 1024. The latent-action branch uses 224×224 inputs, producing a 16×16 final-layer patch grid and 16 codes per transition.
LAPA-DINOv3 LA	Meta FAIR; `facebook/dinov3-vitl16-pretrain-lvd1689m` (Siméoni et al., 2026) plus `laq_dinov3.pt` (Nie et al., 2026; Meituan LongCat, 2026)	ViT-L/16, approximately 303M parameters; 24 layers, 16 heads, hidden size 1024. The latent-action branch uses 224×224 inputs, producing a 14×14 final-layer patch grid after excluding CLS and four register tokens, and 16 codes per transition.
DINOv3 FV	Meta FAIR; `facebook/dinov3-vits16-pretrain-lvd1689m` (Siméoni et al., 2026)	ViT-S/16, approximately 21.6M parameters; 12 layers, six heads, hidden size 384. The FV branch uses 224×224 inputs and final-layer continuous features.
V-JEPA 2 FV	Meta FAIR; official V-JEPA 2 release, `vjepa2_vitl16.pt` (Assran et al., 2025)	ViT-L/16 target encoder (the predictor is not loaded), approximately 304M parameters; 24 layers, 16 heads, hidden size 1024, temporal tubelet size two, and rotary position embeddings.
Wan2.2 FV	Alibaba/Wan Team; official Wan2.2 repository (Wan et al., 2025; Wan Team, 2025)	Wan2.2 VAE encoder with 48 latent channels and temporal/spatial compression $4 \times 16 \times 16$. The FV branch encodes frames separately, mapping 224×224 inputs to $14 \times 14 \times 48$ latents.

DINO backbones and latent-action quantizers. Each LAPA-DINO variant pairs the listed DINO backbone with a separately trained LARYBench quantizer, `laq_dinov2.pt` or `laq_dinov3.pt` (Nie et al., 2026; Meituan LongCat, 2026); both quantizers use 32-dimensional quantization with codebook size eight. The DINOv3 FV encoder is the separate ViT-S/16 checkpoint in Table 3.

LA target layout. With an action horizon of $H = 8$, an LA target contains eight code groups:

$$z_t = (c_{t+j})_{j=0}^{7}, \qquad c_u = (c_{u,1}, \ldots, c_{u,K}). \tag{11}$$

LAPA provides $K = 4$ indices per group, while the two LAPA-DINO variants provide $K = 16$. Flattening the eight groups gives the $L_z = HK$ target positions and learnable LA queries used in Section 3.2. These are categorical indices rather than continuous codebook vectors.

LA temporal construction. Both CALVIN and LIBERO use primary-camera transitions eight frames apart ($\delta = 8$). Let I_u denote the primary-camera frame in observation o_u. A code group computed from (I_u, I_{u+8}) is stored at source index u. For a training anchor t with horizon $H = 8$, it retrieves groups for $u = t, \ldots, t + 7$:

$$c_{t+j} = E_{\mathrm{LA}}^{(m)}(I_{t+j}, I_{t+j+8}), \qquad j = 0, \ldots, 7. \tag{12}$$

The final group therefore depends on I_{t+15}. On LIBERO, these code groups are precomputed offline with the same construction and read together with the actions at $t, \ldots, t + 7$. For FV supervision with horizon eight, the future observation is taken at $t+8$. Temporal validity masks exclude anchors without the observations or labels required by the selected target.

Initialization and frozen components. The VLM is initialized from pretrained MiniCPM-V-4.6 and fine-tuned in full with FP32 parameters; the MMDiT prediction head is randomly initialized. All FV encoders run frozen and without gradients, while the projectors that map their features into the MMDiT remain trainable.

A.2 Implementation, Optimization, and Evaluation

Noise-time sampling. The trajectory index t is distinct from the interpolation times τ_a and τ_f of Section 3.3. Each sampled interpolation time follows

$$u \sim \mathrm{Beta}(1.5, 1), \qquad \tau = \frac{0.999 - \min(u, 0.999)}{0.999}. \tag{13}$$

The MMDiT receives a discretized time condition obtained by scaling τ by 1000. Action noise is used on action-routed samples, whereas FV auxiliary samples use the clean demonstrated action chunk and fix $\tau_a = 1$; their feature interpolation time τ_f is sampled.

Action sampling. At inference, the clean-action sampler uses four steps at $\tau_{a,j} = j/4$, $j = 0, \ldots, 3$, following the interpolation convention of Section 3.3 and always remixing the current estimate with the initial Gaussian noise before predicting the next clean chunk.

Training budgets. All reported training budgets and checkpoint labels count optimizer updates. With g devices, b samples per device, and gradient accumulation factor a, the effective batch size in Section 3.1 is $B_{\mathrm{eff}} = gba$. All LIBERO and CALVIN runs use the 60k-update schedule in Table 5; the full budget is its endpoint. The 30k, 40k, and 50k results, including the half-budget action-only baseline in Table 1, are intermediate checkpoints of these runs.

Action objectives. All reported policies predict clean eight-step chunks of seven-dimensional actions, and the loss ℓ_A in Equation (9) follows the action definition of each dataset. On LIBERO, actions are relative: the end-effector position change, the axis-angle rotation change, and a gripper command. The continuous dimensions are normalized with dataset statistics, the gripper remains binary, and ℓ_A is the elementwise L1 error. On CALVIN, actions are relative end-effector (TCP) commands: normalized position changes, Euler-angle changes, and a binary gripper command ($+1$ open, -1 close), each mapped to $[-1, 1]$ by its 1st and 99th percentiles, and ℓ_A is the mean-squared error. At execution, the six continuous outputs are clipped to $[-1, 1]$, converted by the environment to an absolute TCP target, and executed through inverse kinematics on the seven Franka joints. On the DOS W1 real robot, actions are six joint positions and a continuous gripper command with the same percentile mapping, and ℓ_A is the elementwise L1 error. Within each dataset, all LA, FV, and action-only configurations therefore share the same action loss. Validity masks exclude invalid action timesteps, and the loss averages valid timesteps per channel before averaging channels and samples.

Routing and loss weights. Each sample contributes only its selected loss; grouped route means are weighted by their microbatch fractions, with the auxiliary group additionally scaled by λ_g: 0.1 for LAPA, 0.01 for the two LAPA-DINO variants, and 1 for all FV sources. The LA weights are selected by the ablation in Appendix B.4; each target-specific weight is held fixed across all allocations and checkpoints.

Training data. One model is jointly trained on the four LIBERO suites. All LIBERO configurations, including the action-only baseline and the source-matched controls, use the same 1,693 training episodes, comprising 273,465 frames over 40 tasks at 20 Hz (Table 4). CALVIN follows the ABC→D protocol: training uses environments A, B, and C, while evaluation uses environment D. On both benchmarks, LA augments the shared demonstrations with categorical labels, and FV targets are computed from future observations.

Table 4: LIBERO training demonstrations, shared by all configurations. The final column gives the minimum and maximum episode counts across the ten tasks in each suite.

Suite	Tasks	Episodes	Frames	Episodes/task
Spatial	10	432	52,970	35–47
Object	10	454	66,984	42–50
Goal	10	428	52,042	33–50
Long	10	379	101,469	29–49
Total	40	1,693	273,465	—

Optimization settings. We use AdamW with $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 10^{-8}, and gradient-norm clipping at one. Table 5 lists the learning-rate schedule and the other settings shared by the main LIBERO and CALVIN experiments; both learning rates decay to 5% of their peaks.

Table 5: Shared optimization settings for the main LIBERO and CALVIN experiments.

Setting	Value
Action horizon	8
Effective batch size	128
Training schedule	60k optimizer updates; checkpoints at 30k, 40k, 50k, and 60k
MMDiT learning rate	Peak 10^{-4}; final 5×10^{-6}
VLM learning rate	Peak 10^{-5}; final 5×10^{-7}
Warmup	Linear increase from zero to peak over 2k updates
Decay	Cosine from update 2k to update 60k

Control matching. Within each paired LIBERO or CALVIN routing comparison, we hold fixed model initialization, demonstration data, target encoder, action horizon, visual inputs, auxiliary expert, action objective, effective batch size, total optimizer updates, learning-rate schedule, and inference settings; only the routing probability changes. At P100, every training sample follows the action route and the objective contains only the action loss. Each FV configuration has a source-matched P100 control that retains its FV encoder, MMDiT input f_t, and two-stream architecture. The three LA configurations instead share one action-only baseline, which keeps the two-stream architecture and feeds the $L_z = 32$ learnable LAPA queries to the auxiliary expert. Without an LA loss, these queries are never supervised and carry no target-specific information, so the action expert has no LA signal to exploit, and this baseline serves as the P100 reference for all three LA targets. Cross-family comparisons may additionally differ in target representations and auxiliary experts; the routing-only interpretation applies within a matched configuration.

A.3 Policy Evaluation Protocols and Metrics

LIBERO. Spatial, Object, Goal, and Long contain ten tasks each, with 50 rollouts per task. If $y_{s,j,r}$ indicates success in rollout r of task j in suite s, then

$$\mathrm{SR}_s = \frac{100}{10}\sum_{j=1}^{10}\left(\frac{1}{50}\sum_{r=1}^{50} y_{s,j,r}\right), \qquad \text{Avg.} = \frac{1}{4}\sum_{s=1}^{4}\mathrm{SR}_s.$$

Equal task and rollout counts make Avg. the pooled success rate over 2,000 rollouts; all reported LIBERO values use evaluation seed 0.

CALVIN. All configurations use the same 1,000 ordered five-task sequences and initial states; execution stops at the first failure. CALVIN results in the main text use evaluation seed 0; Table 6 reports results over seeds 0, 1, and 2. Let $L_i \in \{0, \ldots, 5\}$ be the completed prefix length. Then

$$\text{SR-}k = \frac{100}{n}\sum_{i=1}^{n}\mathbf{1}[L_i \geq k], \qquad k = 1, \ldots, 5, \tag{14}$$

and the average task-chain length is

$$\text{Avg. Len.} = \frac{1}{n}\sum_{i=1}^{n} L_i = \frac{1}{100}\sum_{k=1}^{5}(\text{SR-}k). \tag{15}$$

SR-k is cumulative rather than conditional, and Avg. Len. summarizes the same sequence outcomes.

Table 6: CALVIN ABC→D results over evaluation seeds 0, 1, and 2 (mean ± sample standard deviation); SR-5 is in %. Half and Full denote the half- and full-budget action-only baselines. Main-text tables and figures report seed 0.

Target	Route	Avg. Len.	SR-5
Action-only	Half	4.000 ± 0.065	58.8 ± 1.7
	Full	4.335 ± 0.040	71.6 ± 1.8
LAPA	P25	3.501 ± 0.021	45.2 ± 0.9
	P50	3.486 ± 0.059	43.5 ± 2.0
	P75	4.092 ± 0.032	63.0 ± 1.0
LAPA-DINOv2	P25	4.084 ± 0.039	66.6 ± 2.0
	P50	3.916 ± 0.051	63.1 ± 2.0
	P75	4.308 ± 0.052	73.7 ± 1.6
LAPA-DINOv3	P25	4.267 ± 0.016	71.9 ± 0.8
	P50	4.471 ± 0.034	77.8 ± 1.1
	P75	4.395 ± 0.026	75.6 ± 1.4
FV-DINOv3	P25	2.146 ± 0.040	18.1 ± 1.6
	P50	4.365 ± 0.010	76.0 ± 1.0
	P75	4.207 ± 0.053	69.3 ± 2.0
	P100	4.245 ± 0.020	70.1 ± 0.7
FV-V-JEPA 2	P25	4.522 ± 0.026	80.5 ± 0.6
	P50	4.419 ± 0.029	75.0 ± 1.0
	P75	4.239 ± 0.024	72.5 ± 1.0
	P100	4.076 ± 0.031	62.8 ± 0.9
FV-Wan2.2	P25	4.248 ± 0.016	70.6 ± 0.4
	P50	4.254 ± 0.017	70.7 ± 0.3
	P75	4.373 ± 0.010	77.0 ± 0.8
	P100	3.753 ± 0.033	58.9 ± 0.6

B Supplementary Policy Results

B.1 Primary Benchmark Details

Table 7 retains the CALVIN prefix-success metrics omitted from the main comparison table; SR-k and Avg. Len. are defined in Equation (15). In Table 1, Param. reports total model size rounded to the nearest billion. Comparisons in that table do not establish a universal ranking across training protocols.

B.2 Source-Matched FV Controls

Table 8 compares each LIBERO P50 FV configuration at every checkpoint with both the shared action-only baseline and its source-matched control, which differs only in routing (Appendix A.2).

Table 7: CALVIN prefix-success metrics underlying Table 1. SR-1 through SR-5 are cumulative success rates (%); Avg. Len. is measured in tasks per sequence.

Configuration	SR-1	SR-2	SR-3	SR-4	SR-5	Avg. Len.
Action-only (half budget)	98.1	91.5	80.3	69.5	57.7	3.971
Action-only (full budget)	99.4	95.3	87.3	79.1	70.3	4.314
LAPA	96.2	85.0	68.6	54.7	42.1	3.466
LAPA-DINOv2	94.0	86.7	78.6	70.8	63.4	3.935
LAPA-DINOv3	98.2	94.7	89.2	84.3	77.0	4.434
DINOv3 FV	97.4	93.8	88.5	82.2	75.1	4.370
V-JEPA 2 FV	99.0	96.0	89.9	84.2	76.0	4.451
Wan2.2 FV	97.0	92.0	85.2	79.0	70.5	4.237

Table 8: LIBERO FV comparisons with the shared action-only baseline and source-matched controls across checkpoints. Scores are success rates (%); differences are P50 FV minus the indicated reference in percentage points. All columns use the same checkpoint budget. Source-matched P100 controls hold all settings except routing fixed.

FV source	Optimizer updates (k)	Shared action-only	P50 FV	Source-matched P100	Δ shared	Δ source-matched
DINOv3	30	93.20	91.95	96.65	-1.25	-4.70
	40	95.45	96.80	96.95	+1.35	-0.15
	50	96.35	97.55	97.90	+1.20	-0.35
	60	96.50	97.25	97.45	+0.75	-0.20
V-JEPA 2	30	93.20	96.55	96.40	+3.35	+0.15
	40	95.45	97.10	97.55	+1.65	-0.45
	50	96.35	96.50	97.20	+0.15	-0.70
	60	96.50	97.95	96.85	+1.45	+1.10
Wan2.2	30	93.20	97.35	97.15	+4.15	+0.20
	40	95.45	97.55	96.75	+2.10	+0.80
	50	96.35	97.10	97.20	+0.75	-0.10
	60	96.50	98.15	97.55	+1.65	+0.60

B.3 LIBERO P50 Checkpoint Comparisons

Table 9 reports P50 performance at 30k, 40k, 50k, and 60k optimizer updates. These checkpoints track performance through training under the same four-stage evaluation schedule used for the shared action-only baseline in Table 8. Figure 8 plots these LIBERO trajectories, complementing the CALVIN trajectories in Figure 4.

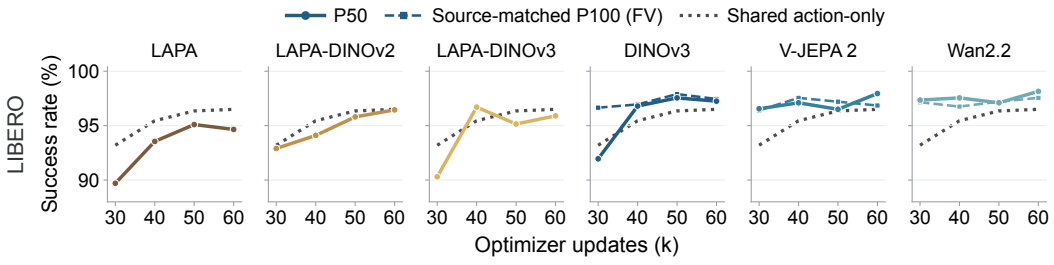

Figure 8: LIBERO performance across budgets at P50, complementing the CALVIN trajectories in Figure 4. Solid: P50; colored dashed: source-matched FV P100; gray dotted: shared action-only baseline, also the LA P100 reference.

Table 9: Detailed LIBERO P50 results at four optimizer-update checkpoints. P50 routes half of the training samples to action prediction in expectation. Success rates are percentages; Avg. is the mean of the four suite rates.

Configuration	Optimizer updates (k)	Spatial	Object	Goal	Long	Avg.
LAPA	30	97.20	98.20	90.20	73.20	89.70
LAPA	40	97.20	98.80	94.20	84.00	93.55
LAPA	50	97.00	98.60	93.40	91.40	95.10
LAPA	60	96.40	98.20	94.20	89.80	94.65
LAPA-DINOv2	30	95.80	97.60	93.60	84.60	92.90
LAPA-DINOv2	40	93.60	97.80	97.40	87.60	94.10
LAPA-DINOv2	50	97.00	98.00	95.80	92.40	95.80
LAPA-DINOv2	60	97.00	99.60	98.00	91.20	96.45
LAPA-DINOv3	30	96.60	97.80	89.20	77.60	90.30
LAPA-DINOv3	40	98.40	99.60	98.00	90.80	96.70
LAPA-DINOv3	50	96.60	97.40	96.40	90.20	95.15
LAPA-DINOv3	60	97.20	97.20	96.80	92.40	95.90
DINOv3 FV	30	94.80	97.00	94.80	81.20	91.95
DINOv3 FV	40	98.00	99.80	97.80	91.60	96.80
DINOv3 FV	50	98.80	99.20	98.80	93.40	97.55
DINOv3 FV	60	98.40	99.20	98.40	93.00	97.25
V-JEPA 2 FV	30	99.00	98.60	97.00	91.60	96.55
V-JEPA 2 FV	40	99.00	99.20	96.80	93.40	97.10
V-JEPA 2 FV	50	98.40	99.40	97.40	90.80	96.50
V-JEPA 2 FV	60	98.80	99.20	98.60	95.20	97.95
Wan2.2 FV	30	99.00	99.80	97.80	92.80	97.35
Wan2.2 FV	40	97.80	99.60	96.60	96.20	97.55
Wan2.2 FV	50	98.60	99.20	96.60	94.00	97.10
Wan2.2 FV	60	98.40	100.00	99.00	95.20	98.15

B.4 LA Loss-Weight Ablation

Table 10 varies the LA loss weight $\lambda_{\text{LA}} \in \{0.01, 0.1, 1.0\}$ for each LA target on LIBERO at P50 and the full budget, with all other settings fixed. For every target, $\lambda_{\text{LA}} = 1.0$ performs worst. The two LAPA-DINO variants perform best at 0.01, whereas LAPA is nearly unchanged between 0.01 and 0.1 and is slightly better at 0.1. We therefore use 0.1 for LAPA and 0.01 for both LAPA-DINO variants in all reported experiments; the same weights are reused on CALVIN without further tuning.

Table 10: LA loss-weight ablation on LIBERO (P50, full budget). Entries are average success rates (Avg., %). Bold: the weight used for each target in all reported experiments.

λ_{LA}	LAPA	LAPA-DINOv2	LAPA-DINOv3
0.01	94.55	**96.45**	**95.90**
0.1	**94.65**	94.95	95.50
1.0	91.95	93.95	94.90

B.5 CALVIN Checkpoint Comparisons

Table 11 reports the absolute P50 scores underlying Figure 4, enabling direct comparisons between LA and FV at each training budget.

Table 12 gives P50-minus-P100 differences for each FV source and checkpoint, using the same source-matched controls as Figure 4.

Table 11: CALVIN average task-chain length (Avg. Len.) under P50 routing at each tested training budget. These absolute scores underlie the P50 trajectories in Figure 4.

Target	Optimizer updates (k)			
	30	40	50	60
Latent action (LA)				
LAPA	3.019	3.313	3.580	3.466
LAPA-DINOv2	3.120	4.194	3.812	3.935
LAPA-DINOv3	3.830	4.157	4.322	4.434
Future vision (FV)				
DINOv3	4.085	4.014	4.333	4.370
V-JEPA 2	4.087	4.430	4.268	4.451
Wan2.2	3.876	3.997	4.094	4.237

Table 12: CALVIN P50 minus source-matched P100 at all four reported checkpoints. Δ Avg. Len. is in tasks per sequence; Δ SR-5 is in percentage points. Each comparison is positive on both metrics.

FV source	Optimizer updates (k)	Δ Avg. Len.	Δ SR-5 (pp)
DINOv3	30	+0.243	+12.9
	40	+0.235	+9.7
	50	+0.270	+11.9
	60	+0.146	+5.0
V-JEPA 2	30	+0.770	+18.2
	40	+0.622	+19.0
	50	+0.258	+10.7
	60	+0.360	+12.8
Wan2.2	30	+0.496	+12.4
	40	+0.702	+17.4
	50	+0.589	+16.2
	60	+0.493	+12.2

Table 13: Tested supervision allocations at the common CALVIN endpoint of 60k optimizer updates. Each cell reports Avg. Len. / SR-5 (%). P25, P50, P75, and P100 denote action-routing probabilities $p_A = 0.25, 0.50, 0.75$, and 1.00, respectively; lower P assigns a larger expected fraction of samples to auxiliary (LA/FV) prediction. FV P100 entries are source-matched controls; LA shares the full-budget action-only baseline in Table 7, hence the dashes in its P100 column. Bold marks the best tested value for each metric within each row, including ties.

Target	P25	P50	P75	P100
Latent action (LA)				
LAPA	3.511 / 45.7	3.466 / 42.1	**4.128 / 63.9**	—
LAPA-DINOv2	4.126 / 68.8	3.935 / 63.4	**4.264 / 72.8**	—
LAPA-DINOv3	4.283 / 72.8	**4.434 / 77.0**	4.421 / **77.0**	—
Future vision (FV)				
DINOv3	2.107 / 16.6	**4.370 / 75.1**	4.193 / 68.7	4.224 / 70.1
V-JEPA 2	**4.522 / 80.0**	4.451 / 76.0	4.240 / 72.1	4.091 / 63.2
Wan2.2	4.264 / 71.1	4.237 / 70.5	**4.377 / 77.4**	3.744 / 58.3

B.6 CALVIN Allocation Comparisons

Table 13 reports every tested allocation at 60k updates: P25–P75 for each LA target and P25–P100 for each FV source. LA uses the shared action-only baseline as P100, reported in Table 7; FV uses source-matched P100 controls.

Table 14 gives the corresponding FV grid at 40k updates; Figure 9 covers all targets, allocations, and budgets.

Table 14: FV supervision allocation at a fixed CALVIN training budget of 40k optimizer updates. P25, P50, P75, and P100 denote action-routing probabilities 0.25, 0.50, 0.75, and 1.00, respectively. P100 is the source-matched control. Batch size and optimization settings are fixed within each source. Bold marks the best tested value for each metric within each row.

	P25		P50		P75		P100	
FV source	Avg. Len.	SR-5 (%)	Avg. Len.	SR-5 (%)	Avg. Len.	SR-5 (%)	Avg. Len.	SR-5 (%)
DINOv3	1.666	10.3	4.014	64.1	**4.223**	**69.7**	3.779	54.4
V-JEPA 2	4.318	73.7	**4.430**	**75.0**	4.384	73.7	3.808	56.0
Wan2.2	4.112	64.4	3.997	64.3	**4.443**	**80.0**	3.295	46.9

B.7 Joint Allocation and Budget Sweeps

Figure 9 reports all three mixed allocations at each tested budget on both benchmarks. LA uses the shared action-only baseline at the same checkpoint; FV uses source-matched P100 controls. This full sweep complements the fixed-60k allocation comparison in Figure 3 and the fixed-P50 budget comparisons in Figures 4 and 8. The plotted gains describe improvements within each configuration; absolute scores and both reference types in the main figures support comparisons across configurations.

B.8 Real-Robot Evaluation Details

Table 15 contains the real-robot values summarized in Figure 5.

The real-robot platform is the DOS W1 left follower arm, equipped with third-person and wrist-mounted RealSense D435i cameras. The RGB input resolutions are 448×252 and 448×336 pixels, respectively, both at 15 Hz. Actions comprise six absolute joint positions and one continuous gripper command (action objectives in Appendix A.2); depth and effort are discarded during preprocessing.

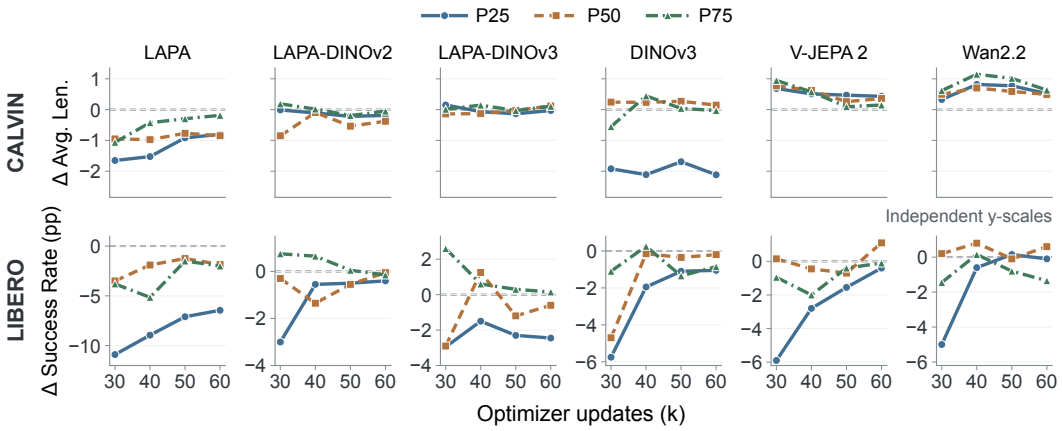

Figure 9: Full sweep over supervision allocation and training budget. Each curve reports gains over the same-stage P100 control: the shared action-only baseline for LA and the source-matched control for FV. Rows show CALVIN Δ Avg. Len. and LIBERO Δ success rate (pp). Dashed zero lines indicate parity with the corresponding control. CALVIN panels share a vertical scale; LIBERO panels use target-specific vertical scales to show each trajectory.

The shared four-task training set contains 203 demonstrations: 50 each for Pick-out, Place-in, and Rearrange-objects, and 53 for Rearrange-cups. The four task instructions are:

1. *Pick the objects out of the basket.*
2. *Place the objects into the basket.*
3. *Rearrange the cups.*
4. *Rearrange the objects.*

Auxiliary configurations use P50. All configurations are trained for 30k optimizer updates and evaluated over 20 trials per task. Each trial has a maximum score of eight. Pick-out, Place-in, and Rearrange-objects contain four objects. For each object, a successful pick contributes one point and a successful placement according to the task instruction contributes one point. Rearrange-cups contains three cups: the same pick-and-place rule contributes up to six points, and the remaining two points evaluate whether the final arrangement is orderly and whether the cup handles face outward from the shelf. These components sum to the trial score, so the maximum accumulated score over 20 trials is 160 for every task. The component-level rule allows partial credit. No scene damage occurred in the reported trials. Results are reported separately for each task.

Table 15: Real-robot manipulation outcomes on the DOS W1 left arm. Auxiliary configurations use P50; all configurations are trained for 30k optimizer updates. Scores accumulate over 20 trials per task (maximum 160; scoring in Appendix B.8). Bold marks the highest score in each task.

Model	Pick-out ($n = 20$)	Place-in ($n = 20$)	Rearrange-cups ($n = 20$)	Rearrange-objects ($n = 20$)
Action-only				
Action-only baseline	116 / 160	92 / 160	80 / 160	124 / 160
Latent action (LA)				
LAPA (P50)	92 / 160	92 / 160	80 / 160	96 / 160
LAPA-DINOv2 (P50)	112 / 160	80 / 160	76 / 160	120 / 160
LAPA-DINOv3 (P50)	106 / 160	88 / 160	72 / 160	92 / 160
Future vision (FV)				
DINOv3 (P50)	138 / 160	100 / 160	96 / 160	132 / 160
V-JEPA 2 (P50)	150 / 160	108 / 160	100 / 160	144 / 160
Wan2.2 (P50)	**156 / 160**	**112 / 160**	**108 / 160**	**148 / 160**

C Diagnostic Protocols and Supplementary Results

This section specifies the diagnostic targets and metrics, then reports the measurements omitted from the main text.

C.1 Diagnostic Protocols and Readout Targets

Regression readouts and data. The regression splits for both datasets are taken from LARY-Bench (Nie et al., 2026; Meituan LongCat, 2026), which re-segments AgiBotWorld-Beta (AgiBot-World-Contributors et al., 2025) and CALVIN (Mees et al., 2022) into image-pair regression samples. The six frozen source representations come from the target encoders and checkpoints in Table 3. Each source representation–readout target pair receives a separate MLP-ResNet (width 4096, two residual blocks, 50 epochs, AdamW). LAPA-style regressions use flattened post-quantization vectors. The main ABS/delta experiment is visual-only with frame-pair stride $s = 45$; AgiBotWorld-Beta ABS predicts all $45 \times 16 = 720$ action components. For V-JEPA 2, each sample supplies only the two endpoint frames of its segment, resized to a short side of 256, center-cropped to 224×224, and ImageNet-normalized; the target encoder returns 196 tokens of dimension 1024.

AgiBotWorld-Beta uses three regression splits. Seen training contains 45-step segments from 169 seen task IDs and is used to fit the readout head. Seen validation contains held-out segments from the same 169 tasks. It selects each probe's checkpoint by validation MSE and also serves as the analysis split for the geometry, clustering, and delta-step5 measurements (Appendix C.2). Geometry and clustering involve no training on this split; because the delta-step5 probe is also selected on it, the delta-step5 NMSE is a descriptive within-seen-task measure used to compare encoders rather than a held-out estimate. Seen validation is not episode-disjoint from seen training: the two splits share 12,239 episodes, although no identical 45-step action segment appears in both. It therefore measures generalization within seen tasks rather than to new trajectories. The unseen split contains 19 further tasks whose task IDs and episodes do not overlap with the seen splits. The reported unseen errors (Table 2) always come from the checkpoint selected on seen validation, never from the best unseen epoch. Table 16 lists the split sizes; these image-pair samples are not independent robot trajectories. CALVIN uses the `train_stride5` and `val_stride5` diagnostic splits and reports errors on the environment-D validation split; it has no seen/unseen task partition. These frozen-source readouts characterize information recoverable by the tested probes.

ABS and delta targets. ABS uses stored absolute trajectories; delta targets Δ_c are coordinate-wise action differences, so their positional components are Cartesian displacements. Each probe is trained on its own transformed labels. The diagnostic setting c is distinct from the policy target offset δ in Section 3.1. Here, $c = 1$ uses adjacent-action differences, $c = 9$ uses action differences with stride nine, and $c = 45$ denotes one difference between the segment's final and initial actions.

Table 16: AgiBotWorld-Beta regression splits derived from LARYBench. Counts are image-pair samples, not trajectories.

Split	Task IDs	Samples
Seen training	169	84,162
Seen validation	169	28,054
Unseen	19	12,303

Each delta setting is a separately trained target, not a metric applied to one ABS predictor. Encoder comparisons are made within each dataset–target column.

The separate delta-step5 task. For a 45-action segment $(\mathbf{a}_0, \ldots, \mathbf{a}_{44})$, let

$$I_5 = (i_0, \ldots, i_9) = (0, 5, 10, 15, 20, 25, 30, 35, 40, 44), \qquad \mathbf{d}_j = \mathbf{a}_{i_j} - \mathbf{a}_{i_{j-1}}, \quad j = 1, \ldots, 9.$$

Each $\mathbf{d}_j \in \mathbb{R}^{16}$. The regression target is

$$\mathbf{y}_{\text{step5}} = \text{vec}(\mathbf{d}_1, \ldots, \mathbf{d}_9) \in \mathbb{R}^{144}.$$

No temporal averaging is applied to this regression target. The last difference spans four action steps ($40 \to 44$), unlike the preceding five-step differences.

Normalized MSE. Statistics are computed from transformed training targets. For AgiBotWorld-Beta, they are estimated per robot type and action dimension; all nine delta-step5 positions share a dimension's statistics. Over valid scalar entries,

$$\text{NMSE} = \frac{1}{|\mathcal{V}|} \sum_{(n,j,d) \in \mathcal{V}} \left(\widehat{\widetilde{y}}_{n,j,d} - \widetilde{y}_{n,j,d}\right)^2,$$

where tildes denote standardized coordinates. Delta predictions are not integrated, and comparisons remain within each dataset–target column.

Millimeter position error. After denormalization, delta outputs are reconstructed from the ground-truth initial action at endpoints $i_1, \ldots, i_M$:

$$\widehat{\mathbf{a}}_{i_j} = \mathbf{a}_0 + \sum_{k=1}^{j} \widehat{\mathbf{d}}_k.$$

For AgiBotWorld-Beta, the millimeter Euclidean position error averages arms, evaluated times, and samples:

$$E_{\text{mm}} = \frac{1000}{N} \sum_{n=1}^{N} \frac{1}{|\mathcal{T}_n|} \sum_{t \in \mathcal{T}_n} \frac{1}{2} \sum_{e \in \{\text{L,R}\}} \left\| \widehat{\mathbf{p}}_{n,t,e} - \mathbf{p}_{n,t,e} \right\|_2 .$$

For CALVIN, which has a single arm, E_{mm} omits the average over arms. For delta-step5, $\mathcal{T}_n = \{5, 10, 15, 20, 25, 30, 35, 40, 44\}$. The initial action is an evaluation boundary condition, not a probe input; the score is neither coordinate-wise nor a per-increment error. ABS outputs are scored directly, without this ground-truth initial-action condition. Thus, ABS and delta errors evaluate different readout tasks.

Table 2 in the main text reports the resulting position errors for each dataset and readout target.

Geometry. Geometry uses mean-pooled features from 28k AgiBotWorld-Beta seen-validation samples and averages the nine delta-step5 differences:

$$\overline{\mathbf{d}} = \frac{1}{9} \sum_{j=1}^{9} \mathbf{d}_j = \frac{\mathbf{a}_{44} - \mathbf{a}_0}{9}.$$

This 16-dimensional descriptor captures net change, not the time-resolved regression target (which it complements rather than replicates), a unit direction, or per-time velocity. Distances use subspaces of the 16 action channels:

$$\texttt{pos6} = [l_x, l_y, l_z, r_x, r_y, r_z], \qquad \texttt{posgrip8} = [\texttt{pos6}, r_g, l_g],$$

that is, the 3D end-effector displacements of both arms, optionally augmented with both gripper channels. S_0^{dist} and S_0^{trust} use `posgrip8`; clustering uses `pos6`. The eight quaternion channels are excluded: the descriptor subtracts actions componentwise, and quaternions are sign-ambiguous and non-Euclidean, so Euclidean distances between them would introduce spurious differences. For randomly sampled pairs of examples, we compute Euclidean distances between their action descriptors and between their latent representations. The pair-distance score S_0^{dist} is the Pearson correlation between these two sets of distances. Higher correlation indicates more consistent global distance structure across the two spaces.

The trustworthiness score S_0^{trust} measures whether nearest neighbors in latent space are also nearby in action space. It penalizes latent-space neighbors that are not action-space neighbors according to their ranks in action space. The linear positional readout is summarized by R^2_{pos}. Correlations between these scores and delta-step5 NMSE are computed across the six encoders, separately from the sample-pair distance correlations within each encoder.

Clustering. Clustering uses mean-pooled features on seen validation, one sample per episode, and K-means with $K = 200$ feature clusters. Action pseudo-labels are obtained separately by clustering ground-truth delta displacements in `pos6`. Normalized mutual information (NMI) compares feature-cluster assignments with these displacement-derived labels and with dataset task IDs.

C.2 Geometry and Task–Action Clustering

Table 17 expands Figure 6a. Beyond the Pearson correlations in its bottom row, trustworthiness has a Spearman coefficient of -0.93 with delta-step5 NMSE. These descriptive associations involve offline readout error, not policy performance.

Table 17: Feature–action geometry and separate delta-step5 regression on AgiBotWorld-Beta seen validation (protocol in Appendix C.1). Regression reports normalized MSE (NMSE). Bold marks the highest geometry score and lowest regression error. The bottom-row Pearson correlations relate each geometry/readout score to NMSE across $N = 6$ encoders.

Encoder	S_0^{dist}	S_0^{trust}	R^2_{pos}	$\text{NMSE}_{\text{delta-step5}}$
LAPA-DINOv3	**0.375**	**0.763**	**0.354**	**0.807**
LAPA-DINOv2	0.189	0.628	0.190	0.809
V-JEPA 2	-0.030	0.523	0.240	0.858
DINOv3	0.047	0.525	-0.027	0.893
Wan2.2	-0.025	0.511	-0.001	0.959
LAPA	-0.089	0.479	0.004	0.959
Reported Pearson r vs. NMSE	-0.820	-0.812	-0.846	–

Table 18 expands Figure 6b and reports complementary associations with displacement-derived action labels and task IDs.

Table 18: Cluster associations on AgiBotWorld-Beta seen validation (protocol in Appendix C.1). NMI compares feature clusters with displacement-derived action labels and with task IDs.

Encoder	$\mathrm{NMI}_{\mathrm{act}}$	$\mathrm{NMI}_{\mathrm{task}}$	Task purity
LAPA-DINOv3	0.412	0.221	0.068
LAPA-DINOv2	0.232	0.174	0.045
LAPA	0.143	0.148	0.032
DINOv3	0.183	0.828	0.653
V-JEPA 2	0.191	0.811	0.663
Wan2.2	0.155	0.577	0.370

C.3 Policy Attention Maps

Figures 7 and 10 share one extraction protocol and use the seven full-budget policies of Table 1. The maps are read from the action expert's cross-attention to the VLM conditioning sequence h_t, not from attention inside the frozen target encoders. We use the last quarter of the 16 MMDiT layers (layers 13–16) and average all 12 heads per layer and the four layers with equal weights. Queries are the eight action tokens; the state token is excluded. Attention is taken at the final update of the four-step action sampler ($\tau_a = 3/4$; Appendix A.2) and averaged over three inference noise seeds. The softmax is computed over the full valid conditioning sequence; we then keep the entries on visual tokens and normalize them to a density within each camera view.

Figure 10 applies this protocol to a second LIBERO observation, complementing the main-text example.

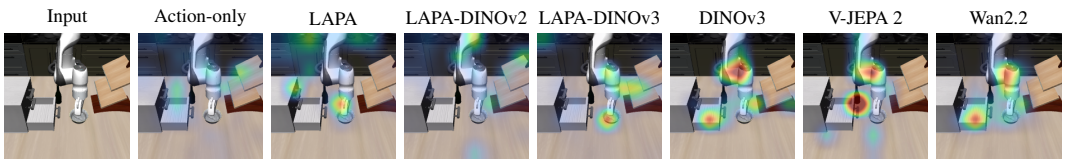

Figure 10: Qualitative policy attention on a second LIBERO observation, complementing Figure 7. Colors show relative intensity within each map.