REVEALING STATE ALIASING IN VISUAL REPRESENTATIONS OF VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language-Action (VLA) models are a promising framework toward general-purpose robot manipulation that leverages the rich prior knowledge of pretrained vision-language models for diverse tasks. However, they often struggle with tasks that require high-precision control. Our analysis across various visual encoders in VLA shows that a key source of the limitation is *state aliasing*, where the visual encoder fails to distinguish fine-grained states that require different actions even though the states are visually similar. Motivated by this observation, we propose a simple yet effective auxiliary inverse dynamics objective for VLA training. By predicting intervening actions between two frames in a demonstration trajectory, our auxiliary task encourages the visual encoder to preserve action-relevant state distinctions. Our approach requires no additional manual annotations and the inverse dynamics prediction is used only during training, introducing no additional computational overhead at inference. Our experiments on LIBERO, CALVIN ABC→D, and SimplerEnv demonstrate consistent improvements in manipulation performance over standard VLA training across diverse VLA architectures. Representation-level analyses further show reduced state aliasing and more accurate action prediction in the learned visual representations, consistent with the trend observed across visual encoders.

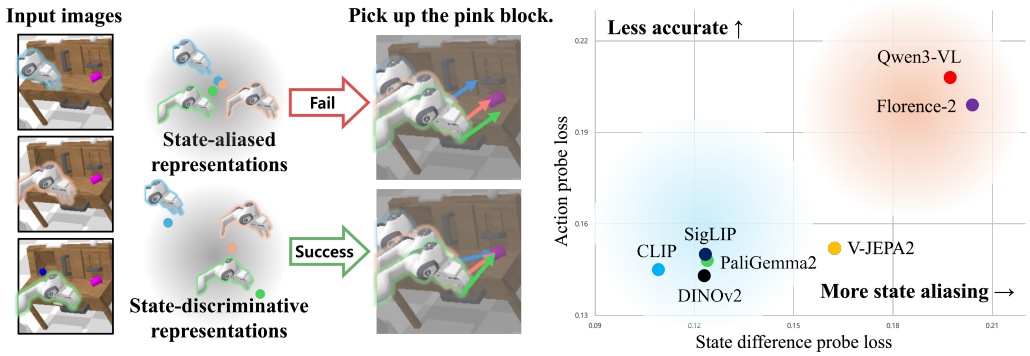
1 INTRODUCTION

Vision-Language-Action (VLA) models (Zitkovich et al., 2023; Black et al., 2025b; Kim et al., 2024b) have emerged as a promising framework for robot manipulation, as they map visual observations and language instructions to executable robot actions while leveraging the semantic priors of pretrained vision-language models (VLMs) (Liu et al., 2023b; Karamcheti et al., 2024; Beyer et al., 2024; Steiner et al., 2024; Xiao et al., 2024; Qwen Team, 2025). Despite this promise, precise manipulation requires visual representations that go beyond high-level vision-language understanding. A policy must distinguish fine-grained state variations, such as subtle changes in the end-effector pose, which can determine the precise motion required to interact with objects in the scene. If such distinctions are lost in the visual representation, the downstream policy may struggle to determine how the robot should move to successfully execute the manipulation.

Despite their importance for manipulation, how well the visual representations used in VLA models preserve fine-grained state distinctions remains underexplored. We refer to the loss of such distinctions as *state aliasing* and analyze this phenomenon across a range of pretrained visual encoders relevant to VLA models (Radford et al., 2021; Zhai et al., 2023; Oquab et al., 2025; Xiao et al., 2024; Steiner et al., 2024; Qwen Team, 2025; Assran et al., 2025). Across encoders, we observe a consistent trend: representations with less state aliasing also support more accurate action prediction. This association suggests that state discrimination is closely related to the utility of visual representations for low-level control.

Since pretrained visual representations are often further adapted during VLA training, we ask whether this process can be guided to mitigate state aliasing. To this end, we introduce **inverse dynamics learning** as an auxiliary objective that directly supervises the VLA vision encoder during training. Given current and future observations from a demonstration trajectory, a lightweight inverse dynamics head predicts the corresponding robot action from their visual features. This objective encourages the encoder to retain visual distinctions that determine low-level robot actions.

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.



(a) State aliasing in visual representations **(b) Association with action prediction**

Figure 1: **State aliasing and its association with action prediction.** (a) State aliasing occurs when fine-grained distinctions between robot states are not sufficiently preserved in the visual representation, which can make the actions required for different states difficult to distinguish. State-discriminative representations instead preserve these distinctions and support more precise action selection. (b) Across frozen visual encoders, representations with lower state-difference probe loss generally also exhibit lower action probe loss. The shaded regions highlight this observed tendency, while individual encoders may exhibit intermediate behavior.

The auxiliary head is used only during training and removed afterward, leaving the original VLA architecture and inference process unchanged.

Experiments on LIBERO (Liu et al., 2023a), CALVIN ABC→D (Mees et al., 2022), and SimplerEnv (Li et al., 2024c) show that auxiliary inverse dynamics improves end-to-end manipulation performance across diverse VLA architectures and evaluation settings, including precise manipulation, long-horizon manipulation, and real-to-sim evaluation. On LIBERO, representation-level analyses further show reduced state aliasing and more accurate action prediction, while complementary analysis on CALVIN shows stronger state-feature alignment. These results are consistent with the relationship observed across pretrained visual encoders, where stronger state discrimination is associated with more accurate action prediction.

Our contributions are as follows:

- We analyze state aliasing in visual representations relevant to VLA models and show that, across pretrained visual encoders, stronger fine-grained state discrimination is associated with more accurate action prediction.
- We introduce auxiliary inverse dynamics learning as a training-time objective for mitigating state aliasing in VLA vision encoders, requiring no additional annotations or changes to the inference pipeline.
- We validate the approach across five VLA configurations and three manipulation benchmarks, showing improvements in end-to-end policy performance. Representation-level analyses further reveal reduced state aliasing and more accurate action prediction on LIBERO, together with stronger state-feature alignment on CALVIN.

2 Related Work

2.1 Vision-Language-Action Modeling

Vision-language-action (VLA) models adapt pretrained vision-language models (VLMs) (Liu et al., 2023b; Karamcheti et al., 2024; Beyer et al., 2024; Qwen Team, 2025) to map visual observations and language instructions to low-level robot actions, establishing VLM initialization as a central design choice for generalist robot policies (Zitkovich et al., 2023; Black et al., 2025b; Kim et al., 2024b; Yang et al., 2025). Recent studies have explored diverse policy formulations, including autoregressive token prediction (Zitkovich et al., 2023; Kim et al., 2024b; Wang et al., 2026b), con-

tinuous action prediction via direct regression, diffusion, or flow matching (Black et al., 2025b; Li et al., 2024a; Wen et al., 2025), spatially grounded action prediction (Qu et al., 2025; Li et al., 2025), and hybrid discrete-continuous action modeling (Liu et al., 2026; Driess et al., 2025). VLA training has also been augmented with auxiliary prediction, grounding, and reasoning objectives. Some approaches predict future visual observations to provide spatial and temporal supervision beyond action prediction (Wu et al., 2024; Zitkovich et al., 2023; Zhao et al., 2025; Zhang et al., 2025a;b), while others introduce intermediate objectives such as point or trace prediction and chain-of-thought, latent, or grounded reasoning prior to action generation (Niu et al., 2024; Yuan et al., 2026; Yang et al., 2025; Zawalski et al., 2024; Huang et al., 2025a; Yang et al., 2026b; Sun et al., 2025; Black et al., 2025a; Huang et al., 2025b; 2026). Other methods further use robot or environment state information as additional supervision (Kim et al., 2026; Dong et al., 2026). Our work instead studies state aliasing as a limitation of VLA visual representations and examines its relationship with action prediction.

2.2 Inverse dynamics modeling

Inverse dynamics predicts the action between two observations and has been used in several learning-based robotics settings. In predictive policy learning, it maps predicted future visual states to executable actions, thereby connecting visual prediction or planning with low-level robot control (Du et al., 2023; Tian et al., 2025; Hu et al., 2025; Zhang et al., 2026b). Another line of work uses inverse dynamics to recover actions or latent actions from action-free videos, turning observed state transitions into supervisory signals for subsequent policy learning without explicit action labels (Schmidt & Jiang, 2024; Ye et al., 2025; Yang et al., 2026a; Routray et al., 2026). Closest to our work, inverse dynamics has been studied for imitation learning and representation learning, where predicting actions from state transitions helps preserve control-relevant information and encourages representations to retain distinctions useful for downstream control (Dasari & Gupta, 2021; Brandfonbrener et al., 2023; Li et al., 2024b; Kim et al., 2024a; Cui et al., 2024; Shi et al., 2026). Our approach specifically targets state aliasing in VLA visual representations, using auxiliary inverse dynamics during end-to-end policy learning to encourage the vision encoder to preserve fine-grained state distinctions relevant to action prediction.

3 Diagnostic: State Aliasing in Visual Representations

We define *state aliasing* as the failure of visual encoders to preserve fine-grained distinctions between robot states. To quantify this, we introduce a simple probing protocol. We first freeze each visual encoder and then train a lightweight MLP to predict state differences between nearby observations. Lower prediction error indicates that a visual encoder better preserves the state distinctions, i.e., suffers less from state aliasing, which potentially enables more precise control. We apply this diagnostic across diverse visual encoders and examine its relationship with action prediction.

3.1 Probing Setup

We conduct our analysis on LIBERO (Liu et al., 2023a), covering 40 manipulation tasks across its four suites. We consider seven visual encoders: CLIP (Radford et al., 2021), SigLIP (Zhai et al., 2023), DINOv2 (Oquab et al., 2025), V-JEPA2 (Assran et al., 2025), and the vision encoders of Qwen3-VL-8B (Qwen Team, 2025), PaliGemma 2 3B (Steiner et al., 2024), and Florence-2 (Xiao et al., 2024). These models provide a diverse set of visual representations for our analysis. For each encoder, we freeze all encoder parameters and train lightweight MLPs on top of the resulting visual features. We evaluate representations through two complementary tasks: robot state-difference prediction and action prediction.

State-difference probe. We construct pairs of nearby observations (o_i, o_j) from the same demonstration trajectory, with j sampled within 1–10 stored frames after i, and use the same fixed observation pairs for all visual encoders. We encode each observation independently with a frozen visual encoder f and compute the difference between their visual features. We then train a lightweight MLP g_s to predict the corresponding robot state difference:

$$\Delta z_{ij} = f(o_j) - f(o_i), \qquad \widehat{\Delta s}_{ij} = g_s(\Delta z_{ij}), \tag{1}$$

Table 1: Frozen-encoder diagnostic results on LIBERO. Visual encoders are ordered by state-difference probe loss. "State Diff." denotes state-difference probe loss, and "Action" denotes action probe loss. Lower is better.

Visual Encoder	State Diff. ↓	Action ↓
CLIP	0.109	0.147
DINOv2	0.123	0.144
SigLIP	0.123	0.149
PaliGemma 2 3B	0.124	0.147
V-JEPA2	0.162	0.150
Qwen3-VL-8B	0.197	0.208
Florence-2	0.204	0.199

where Δs_{ij} denotes the normalized difference between the robot states at the two observations. Each probe is trained for 5K optimization steps, and we report the normalized MSE. Details of pair sampling, state normalization, and probe architecture are provided in Appendix C.2.

Action probe. The action probe takes only the visual tokens from a single current observation, without language instructions. Since task identity is not provided to the probe, we train a separate lightweight MLP for each of the 40 LIBERO tasks. Each MLP predicts four actions, each with 7 DoF, and is trained for 1K optimization steps. We report the training loss averaged across all 40 tasks. Additional training details are provided in Appendix C.3.

3.2 Correlation with Action Prediction

We next examine whether state aliasing is related to how well a visual representation supports action prediction. Figure 1 (b) visualizes the relationship between state-difference and action probe losses, while Table 1 reports the corresponding probe results for each encoder. We observe a consistent trend: encoders with lower state-difference probe loss generally also achieve lower action probe loss. This association is also supported by their rank ordering (Spearman $\rho = 0.89$), suggesting that representations that better preserve fine-grained distinctions between robot states tend to provide more useful features for predicting low-level robot actions. Additional policy rollout evaluations of the trained action probe MLPs are provided in Appendix C.4.

Since pretrained visual representations are often further adapted during VLA training, these observations motivate us to ask whether this adaptation can be explicitly guided to mitigate state aliasing and whether such mitigation also improves end-to-end VLA performance.

4 Method

4.1 Problem Formulation

We consider a language-conditioned robot manipulation dataset $\mathcal{D} = \{\tau_i\}_{i=1}^{N}$ consisting of demonstration trajectories. Each trajectory is represented as

$$\tau = \left(\ell, \{(o_t, a_t)\}_{t=1}^{T}\right), \tag{2}$$

where ℓ denotes a language instruction, o_t a visual observation, and a_t the corresponding robot action. Given the current observation and language instruction, a VLA policy π_θ predicts an action chunk,

$$\hat{a}_{t:t+H} = \pi_\theta(o_t, \ell), \tag{3}$$

and is trained with its original VLA objective $\mathcal{L}_{\text{VLA}}\left(\hat{a}_{t:t+H}, a_{t:t+H}\right)$.

Under standard VLA training, the vision encoder is optimized through action supervision, which does not explicitly encourage it to preserve fine-grained state distinctions. We hypothesize that this

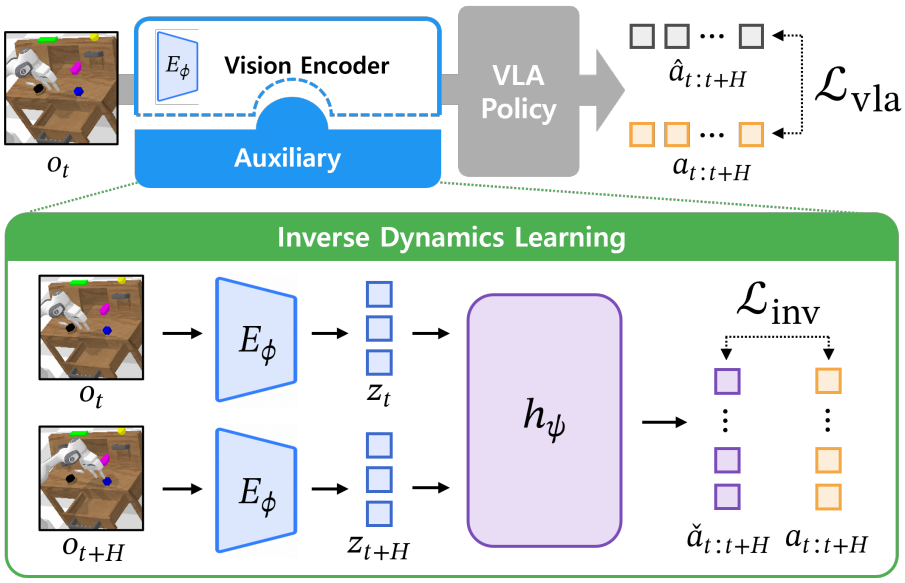


Figure 2: **Method overview.** To mitigate state aliasing during VLA training, we augment the original policy objective with auxiliary inverse dynamics supervision. Given the current observation o_t and a future observation o_{t+H} from the same demonstration trajectory, the shared vision encoder E_ϕ independently produces z_t and z_{t+H}, which are used by the inverse dynamics head h_ψ to predict the intervening action chunk $\check{a}_{t:t+H}$ under $\mathcal{L}_{\text{inv}}$. This provides direct supervision for preserving fine-grained state distinctions in the VLA visual representation. The auxiliary branch is removed after training, leaving the original VLA architecture and inference process unchanged.

can leave room for state aliasing when nearby states receive similar action targets; indeed, we find that many locally similar states in the CALVIN ABC (Mees et al., 2022) training data are associated with highly similar action supervision (Appendix B.1). Motivated by the relationship observed in Sec. 3.2, we therefore introduce an auxiliary objective that directly supervises the vision encoder to preserve fine-grained state distinctions.

4.2 Inverse Dynamics Supervision

Inverse dynamics is the task of predicting the action between two observations. We use this task as an auxiliary supervision signal for the VLA vision encoder, encouraging it to preserve visual distinctions that are relevant to low-level robot actions. Given the current observation o_t and a future observation o_{t+H} from the same demonstration trajectory, the vision encoder E_ϕ independently encodes the two observations as

$$z_t = E_\phi(o_t), \qquad z_{t+H} = E_\phi(o_{t+H}). \tag{4}$$

A lightweight inverse dynamics head h_ψ takes both visual features and predicts the intermediate action chunk:

$$\check{a}_{t:t+H} = h_\psi(z_t, z_{t+H}), \tag{5}$$

where $\check{a}_{t:t+H}$ denotes the inverse dynamics prediction. We use the same horizon H as the VLA action prediction horizon. Gradients from this auxiliary objective are propagated through the inverse dynamics head into the vision encoder, directly shaping the visual representation.

The auxiliary inverse dynamics loss is defined as

$$\mathcal{L}_{\text{inv}} = \mathcal{L}_{\text{motion}}\left(\check{a}_{t:t+H}, a_{t:t+H}\right) + \lambda_g \mathcal{L}_{\text{gripper}}\left(\check{a}_{t:t+H}, a_{t:t+H}\right), \tag{6}$$

where $\mathcal{L}_{\text{motion}}$ is an ℓ_1 loss on the end-effector motion, $\mathcal{L}_{\text{gripper}}$ is a binary cross-entropy loss on the gripper action, and λ_g balances the two terms. Since the observation pairs and intermediate actions are obtained directly from the original demonstration trajectories, the auxiliary objective requires no additional annotations. Details of the inverse dynamics head and other implementation choices are provided in the Appendix A.1.

Pseudo time reversal augmentation. Because robot demonstrations contain strong correlations between visual states and the actions taken from them, the vision encoder may learn visual features that implicitly encode action-specific shortcuts (Xing et al., 2025) rather than the underlying state distinctions. To reduce this risk, we additionally apply a stochastic pseudo-reversed augmentation for auxiliary inverse dynamics learning by swapping the order of the observation pair and reversing the corresponding action sequence with negated motion offsets. The exact construction is provided in the Appendix A.2.

4.3 Overall Pipeline

We combine the standard VLA objective with the auxiliary inverse dynamics objective as

$$\mathcal{L} = \mathcal{L}_{\text{VLA}} + \lambda_{\text{inv}} \mathcal{L}_{\text{inv}}, \tag{7}$$

where λ_{inv} controls the strength of the auxiliary supervision. The inverse dynamics head is used only during training and removed afterward. At inference time, the original VLA architecture and inference process remain unchanged.

5 Experiments

We evaluate our method from three complementary perspectives. First, on LIBERO (Liu et al., 2023a), we examine how auxiliary inverse dynamics affects the learned VLA visual representation using state-difference and action probes and measure end-to-end VLA performance after downstream fine-tuning. Second, we test the generality of our method across different VLA architectures and benchmarks on CALVIN ABC→D (Mees et al., 2022) and SimplerEnv (Li et al., 2024c). Finally, we perform state-feature alignment analysis on CALVIN to provide an additional view of how auxiliary inverse dynamics changes the learned visual representations.

5.1 Experimental Setup

Baseline models. We evaluate our method across five VLA configurations with diverse architectures and action modeling strategies. VLM4VLA (Zhang et al., 2026a) is built from a pretrained VLM with minimal architectural modification, making it suitable for analyzing how auxiliary inverse dynamics affects the learned visual representation. FLOWER (Reuss et al., 2025) is a compact VLA model that predicts actions with flow matching. GR-1 (Wu et al., 2024) additionally incorporates future-frame prediction during policy learning, allowing us to examine whether auxiliary inverse dynamics remains effective alongside an existing predictive objective. SpatialVLA (Qu et al., 2025) predicts spatial action tokens through explicit 3D spatial representations, providing a structurally distinct VLA architecture for evaluation. We use the 8B variant of VLM4VLA for the LIBERO analysis, evaluate both the 8B and 2B variants of VLM4VLA together with FLOWER and GR-1 on CALVIN, and evaluate SpatialVLA on SimplerEnv. For CALVIN and SimplerEnv, we additionally report results from prior VLA methods as references to provide broader context.

Training protocol. For each baseline, we follow its original training pipeline and schedule while adding auxiliary inverse dynamics supervision to the vision encoder. The auxiliary hyperparameters are configured separately for each model, with detailed settings provided in Appendices D–F. For each policy input at time t, the auxiliary objective additionally uses the future observation at $t + H$ from the same trajectory and predicts the intervening action chunk. We evaluate the final checkpoint obtained at the end of each training schedule. For FLOWER, the vanilla baseline reported in the main paper is taken from the original work, while the +Ours variant is trained using the released configuration augmented with our auxiliary objective.

Table 2: Evaluation of VLA visual representations on LIBERO after LIBERO-90 pretraining. We report state-difference and action probe losses using frozen vision encoders, together with rollout success rates of the trained action probe MLPs. Lower probe loss and higher success rate are better.

	Probe Loss ↓		Success Rate (%) ↑				
Vision Encoder	State Diff.	Action	Spatial	Object	Goal	Long	Avg.
Qwen3-VL-8B	0.197	0.208	49.5	39.5	43.0	9.5	35.4
VLM4VLA	0.163	0.152	**57.0**	39.5	55.0	18.0	42.4
+ Ours	**0.118**	**0.137**	**57.0**	**43.5** (+4.0)	**63.0** (+8.0)	**19.5** (+1.5)	**45.8** (+3.4)

Table 3: End-to-end manipulation performance on LIBERO after downstream fine-tuning. We report success rates on the four LIBERO suites and their average, evaluated with 20 rollouts per task (200 rollouts per suite; 800 rollouts in total).

Method	Success Rate (%) ↑				
	Spatial	Object	Goal	Long	Average
VLM4VLA	55.0	84.0	50.0	26.5	53.9
+ Ours	**94.5** (+39.5)	**90.5** (+6.5)	**73.5** (+23.5)	**77.0** (+50.5)	**83.9** (+30.0)

5.2 State Aliasing and Policy Performance on LIBERO

LIBERO is a manipulation benchmark with diverse tasks that often require precise object grasping and placement. It provides LIBERO-90 as a pretraining set, together with four disjoint downstream suites (Spatial, Object, Goal, and Long) for fine-tuning and evaluation. To assess state aliasing after VLA training, we train vanilla and +Ours VLM4VLA models on LIBERO-90, starting from the pretrained Qwen3-VL-8B (Qwen Team, 2025) initialization used by VLM4VLA. We then freeze their vision encoders and evaluate them on the training data from the four downstream LIBERO suites using the state-difference and action probes introduced in Sec. 2. As a complementary evaluation, we also roll out the trained action probe MLPs as policies and measure their success rates. Finally, we fine-tune the vanilla and +Ours VLA models pretrained on LIBERO-90 using the combined training data from the four LIBERO suites and evaluate their end-to-end manipulation performance.

Table 2 summarizes the representation-level results. The state-difference probe loss decreases from 0.197 for the pretrained Qwen3-VL-8B encoder to 0.163 after vanilla VLA training and further to 0.118 with auxiliary inverse dynamics. A similar trend appears in action prediction: the action probe loss decreases from 0.208 for the pretrained encoder to 0.152 after vanilla VLA training and 0.137 with our method. The lower state-difference probe loss indicates that auxiliary inverse dynamics further mitigates state aliasing beyond vanilla VLA training. The accompanying reduction in action probe loss is consistent with the relationship observed across pretrained visual encoders in Sec. 2. As a complementary evaluation, the average rollout success rate of the action probe MLPs increases from 42.4% after vanilla VLA training to 45.8% with auxiliary inverse dynamics.

We further evaluate end-to-end VLA performance after downstream fine-tuning. As shown in Table 3, auxiliary inverse dynamics improves the average success rate from 53.9% to 83.9% (+30.0 pp). These results show that auxiliary inverse dynamics also improves downstream end-to-end VLA performance on LIBERO.

5.3 Generality across VLA Architectures and Benchmarks

CALVIN ABC→D. CALVIN is a language-conditioned long-horizon manipulation benchmark that evaluates the execution of consecutive tasks over long sequences. Following the standard ABC→D protocol, we apply auxiliary inverse dynamics to the 8B and 2B variants of VLM4VLA, together with FLOWER and GR-1, trained on the ABC environments and evaluate the resulting policies in the unseen D environment over 1000 rollout sequences. Tasks are evaluated sequentially, with each new task provided after the previous one is successfully completed. If a task fails,

Table 4: Results on CALVIN ABC→D. Following the standard evaluation protocol, each rollout consists of a sequence of up to five tasks executed consecutively without resetting the environment between tasks. We report the success rates for completing 1, 2, 3, 4, and 5 consecutive tasks, together with the average number of tasks completed per sequence. If a task fails, all subsequent tasks in that sequence are counted as unsuccessful. Reference results are included from prior VLA methods to provide broader context. The Baseline models block compares each baseline with its +Ours variant. For VLM4VLA, 8B and 2B denote variants using Qwen3-VL-8B and Qwen3-VL-2B, respectively. * indicates baseline results reproduced in our environment.

Type	Method	Success Rate (%) ↑					Avg. Len. ↑
		1	2	3	4	5	
Reference	UP-VLA (Zhang et al., 2025a)	92.8	86.5	81.5	76.9	69.9	4.08
	RoboVLMs (Li et al., 2026)	98.0	93.6	85.4	77.8	70.4	4.25
	Seer (Tian et al., 2025)	96.3	91.6	86.1	80.3	74.0	4.28
	VPP (Hu et al., 2025)	96.5	90.9	86.6	82.0	76.9	4.33
	VLA-Adapter (Wang et al., 2026a)	99.1	94.6	88.8	82.8	76.5	4.42
	DreamVLA (Zhang et al., 2025b)	98.2	94.6	89.5	83.4	78.1	4.44
Baseline models	VLM4VLA 8B* (Zhang et al., 2026a)	93.4	85.9	79.7	75.2	69.2	4.03
	+ **Ours**	94.4 (+1.0)	86.7 (+0.8)	81.5 (+1.8)	76.6 (+1.4)	70.5 (+1.3)	4.10 (+0.07)
	VLM4VLA 2B* (Zhang et al., 2026a)	92.7	84.1	74.3	64.0	55.1	3.70
	+ **Ours**	92.8 (+0.1)	84.1	75.7 (+1.4)	67.6 (+3.6)	59.6 (+4.5)	3.80 (+0.10)
	FLOWER (Reuss et al., 2025)	99.3	95.9	90.5	84.8	77.5	4.54
	+ **Ours**	99.5 (+0.2)	96.6 (+0.7)	91.2 (+0.7)	86.9 (+2.1)	81.6 (+4.1)	4.56 (+0.02)
	GR-1* (Wu et al., 2024)	87.9	75.8	65.6	56.4	49.0	3.35
	+ **Ours**	88.5 (+0.6)	76.5 (+0.7)	67.0 (+1.4)	58.9 (+2.5)	49.4 (+0.4)	3.40 (+0.05)

Table 5: Results on the SimplerEnv Bridge benchmark. We report task success rates over 24 trials for each object category and the average success rate across categories. Reference results from prior VLA methods are included to provide broader context. The Baseline models block compares SpatialVLA with its +Ours variant. * indicates baseline results reproduced in our environment.

Type	Method	Success Rate (%) ↑				
		Carrot	Eggplant	Spoon	Cube	Average
Reference	OpenVLA	0.0	0.0	0.0	4.1	1.0
	RoboVLMs (pre-trained)	25.0	0.0	20.8	8.3	13.5
	RoboVLMs (fine-tuned)	25.0	58.3	29.2	12.5	31.3
	SpatialVLA (pre-trained)	20.8	54.2	12.3	20.8	27.0
Baseline models	SpatialVLA* (fine-tuned)	12.5	66.6	25.0	16.7	30.2
	+ **Ours**	12.5	75.0 (+8.4)	25.0	20.8 (+4.1)	33.3 (+3.1)

the remaining tasks are counted as unsuccessful. We report the success rates for completing 1–5 consecutive tasks and the average number of tasks completed per sequence.

As shown in Table 4, auxiliary inverse dynamics improves the corresponding baseline across all evaluated settings. The average completed sequence length increases from 4.03 to 4.10 for VLM4VLA 8B, from 3.70 to 3.80 for VLM4VLA 2B, from 4.54 to 4.56 for FLOWER, and from 3.35 to 3.40 for GR-1. The improvement across both VLM4VLA scales indicates that the benefit of auxiliary inverse dynamics is not specific to a particular VLM scale. The improvement on GR-1 further suggests that auxiliary inverse dynamics is complementary to an existing future-frame prediction objective.

SimplerEnv Bridge. We further evaluate on SimplerEnv, a real-to-sim benchmark that evaluates policies trained on real-world robot data in simulated environments. Following the WidowX Bridge setting (Walke et al., 2023), we initialize SpatialVLA from its released pretrained checkpoint, fine-tune it on Bridge, and evaluate four manipulation tasks over 24 trials each. As shown in Table 5, auxiliary inverse dynamics improves the average success rate from 30.2% to 33.3%. This result shows that auxiliary inverse dynamics also improves a spatially structured VLA in the real-to-sim setting.

Table 6: State-feature alignment on CALVIN. We report pixel-controlled partial Spearman correlations between visual feature distances and normalized 6-DoF end-effector pose distances using 6,000 frame pairs from CALVIN validation trajectories. Higher is better, and bold numbers indicate the best result within each model group.

Model	Partial Spearman $\rho \uparrow$	
	Cosine distance	Scale difference
Qwen3-VL-8B (Qwen Team, 2025)	0.184	0.002
VLM4VLA 8B (Zhang et al., 2026a)	0.489	0.242
+ Ours	**0.600**	**0.309**
GR-1 (pretrained) (Wu et al., 2024)	-0.056	0.033
GR-1	0.303	0.123
+ Ours	**0.600**	**0.189**

Across CALVIN and SimplerEnv, auxiliary inverse dynamics improves manipulation performance across distinct VLA architectures in both long-horizon and real-to-sim settings. Because the auxiliary head is used only during training, these improvements require no changes to the original VLA inference pipeline.

5.4 State-Feature Alignment Analysis

We further analyze state-feature alignment on CALVIN as a complementary view of how VLA training and auxiliary inverse dynamics affect the learned visual representations. Since state aliasing concerns whether fine-grained state distinctions are preserved in the representation, we examine whether changes in robot pose are reflected in the visual feature space beyond low-level pixel differences.

Specifically, we sample frame pairs from the same language-annotated trajectory segment and compute a pixel-controlled partial Spearman correlation between the visual feature distance and the normalized 6-DoF end-effector pose distance, while controlling for low-level pixel differences measured by pixel MSE. We report this correlation using two complementary feature metrics: the cosine distance between mean-pooled visual token features and the scale difference between their feature norms. Higher correlation indicates that changes in robot pose are more strongly reflected in the learned visual representation after accounting for pixel-level variation. Further details are provided in Appendix G.

As shown in Table 6, both model families exhibit the same progression. For VLM4VLA 8B, the correlation based on cosine distance increases from 0.184 for the pretrained Qwen3-VL-8B encoder to 0.489 after vanilla VLA training and further to 0.600 with auxiliary inverse dynamics. The correlation based on feature-scale difference similarly increases from 0.002 to 0.242 and 0.309. GR-1 shows the same pattern: the corresponding correlations increase from -0.056 to 0.303 and 0.600 for mean cosine distance, and from 0.033 to 0.123 and 0.189 for feature-scale difference. These results provide complementary evidence that auxiliary inverse dynamics makes the learned visual representations more sensitive to fine-grained robot pose changes beyond low-level pixel differences, consistent with the representation-level improvements observed on LIBERO.

6 Conclusion

This work studies state aliasing in the visual representations of VLA models and its relationship to action prediction. Across diverse visual encoders, we find that representations with less state aliasing also support more accurate action prediction, motivating auxiliary inverse dynamics as an additional training signal for the VLA vision encoder during policy learning. Across multiple VLA architectures and benchmarks, this objective improves manipulation performance without modifying the original inference pipeline. Representation-level analyses further show reduced state aliasing on LIBERO and stronger state-feature alignment on CALVIN, supporting the role of fine-grained state discrimination in VLA representation learning.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript drafting and editing, literature retrieval and discovery, code development and debugging, and discussion of experimental design and result interpretation. All AI-assisted outputs were reviewed and verified by the authors. The authors take full responsibility for the final content of the paper, including all text, claims, code, and experimental results.

ETHICS STATEMENT

This work uses existing public research datasets and benchmarks, including LIBERO, CALVIN, and BridgeData V2. We use these datasets in accordance with their respective licenses and terms of use. This work does not involve newly collected human-subject data or personally identifiable information.

REPRODUCIBILITY STATEMENT

We provide detailed training configurations, probe architectures, evaluation protocols, and benchmark-specific implementation details in the Appendix. These include the hyperparameters for auxiliary inverse dynamics learning, state-difference and action probes, and the training and evaluation settings for LIBERO, CALVIN, and SimplerEnv.

REFERENCES

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew J. 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.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey A. Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bosnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier J. Hénaff, Xi Xiong, Radu Soricut, Jeremiah Harmsen, and Xiaohua Zhai. PaliGemma: A versatile 3b VLM for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

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 *CoRL*, 2025a.

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 *RSS*, 2025b.

David Brandfonbrener, Ofir Nachum, and Joan Bruna. Inverse dynamics pretraining learns good representations for multitask imitation. In *NeurIPS*, 2023.

Zichen Jeff Cui, Hengkai Pan, Aadhithya Iyer, Siddhant Haldar, and Lerrel Pinto. DynaMo: In-domain dynamics pretraining for visuo-motor control. In *NeurIPS*, 2024.

Sudeep Dasari and Abhinav Gupta. Transformers for one-shot visual imitation. In *CoRL*, 2021.

Jiahua Dong, Yunze Man, Pavel Tokmakov, and Yu-Xiong Wang. Capturing visual environment structure correlates with control performance. In *ICLR*, 2026.

Danny Driess, Jost Tobias Springenberg, Brian Ichter, Lili Yu, Adrian Li-Bell, Karl Pertsch, Allen Z. Ren, Homer Walke, Quan Vuong, Lucy Xiaoyang Shi, and Sergey Levine. Knowledge insulating vision-language-action models: Train fast, run fast, generalize better. In *NeurIPS*, 2025.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. In *NeurIPS*, 2023.

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 *ICML*, 2025.

Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. ThinkAct: Vision-language-action reasoning via reinforced visual latent planning. In *NeurIPS*, 2025a.

Chi-Pin Huang, Yunze Man, Zhiding Yu, Min-Hung Chen, Jan Kautz, Yu-Chiang Frank Wang, and Fu-En Yang. Fast-ThinkAct: Efficient vision-language-action reasoning via verbalizable latent planning. In *CVPR*, 2026.

Wenhui Huang, Changhe Chen, Han Qi, Chen Lv, Yilun Du, and Heng Yang. MoTVLA: A vision-language-action model with unified fast-slow reasoning. *arXiv preprint arXiv:2510.18337*, 2025b.

Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic VLMs: Investigating the design space of visually-conditioned language models. In *ICML*, 2024.

Donghu Kim, Hojoon Lee, Kyungmin Lee, Dongyoon Hwang, and Jaegul Choo. Investigating pre-training objectives for generalization in vision-based reinforcement learning. In *ICML*, 2024a.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Paul 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 *CoRL*, 2024b.

Taeyoung Kim, Jimin Lee, Myungkyu Koo, Dongyoung Kim, Kyungmin Lee, Changyeon Kim, Younggyo Seo, and Jinwoo Shin. Contrastive representation regularization for vision-language-action models. In *ICML*, 2026.

Peiyan Li, Yixiang Chen, Hongtao Wu, Xiao Ma, Xiangnan Wu, Yan Huang, Liang Wang, Tao Kong, and Tieniu Tan. BridgeVLA: Input-output alignment for efficient 3d manipulation learning with vision-language models. In *NeurIPS*, 2025.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. CogACT: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024a.

Siyuan Li, Xun Wang, Rongchang Zuo, Kewu Sun, Lingfei Cui, Jishiyu Ding, Peng Liu, and Zhe Ma. Robust visual imitation learning with inverse dynamics representations. In *AAAI*, 2024b.

Xinghang Li, Peiyan Li, Long Qian, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Xinlong Wang, Di Guo, Tao Kong, Hanbo Zhang, and Huaping Liu. What matters in building vision–language–action models for generalist robots. *Nature Machine Intelligence*, 2026. doi: 10.1038/s42256-025-01168-7.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. In *CoRL*, 2024c.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: benchmarking knowledge transfer for lifelong robot learning. In *NeurIPS*, 2023a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023b.

Jiaming Liu, Hao Chen, Zhuoyang Liu, Pengju An, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, Chengkai Hou, Mengdi Zhao, KC Alex Zhou, Pheng-Ann Heng, and Shanghang Zhang. HybridVLA: Collaborative diffusion and autoregression in a unified vision-language-action model. In *ICLR*, 2026.

Oier Mees, Lukás Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics Autom. Lett.*, 2022.

Dantong Niu, Yuvan Sharma, Giscard Biamby, Jerome Quenum, Yutong Bai, Baifeng Shi, Trevor Darrell, and Roei Herzig. LLARVA: vision-action instruction tuning enhances robot learning. In *CoRL*, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *TMLR*, 2025.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Jiayuan Gu, Zhigang Wang, Yan Ding, Bin Zhao, Dong Wang, and Xuelong Li. SpatialVLA: Exploring Spatial Representations for Visual-Language-Action Models. In *RSS*, 2025.

Qwen Team. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Moritz Reuss, Hongyi Zhou, Marcel Rühle, Ömer Erdinç Yağmurlu, Fabian Otto, and Rudolf Lioutikov. FLOWER: Democratizing generalist robot policies with efficient vision-language-flow models. In *CoRL*, 2025.

Sandeep Routray, Hengkai Pan, Unnat Jain, Shikhar Bahl, and Deepak Pathak. ViPRA: Video prediction for robot actions. In *ICLR*, 2026.

Dominik Schmidt and Minqi Jiang. Learning to act without actions. In *ICLR*, 2024.

Junhao Shi, Siyin Wang, Xiaopeng Yu, Li Ji, Jingjing Gong, and Xipeng Qiu. Learning to move before learning to do: Task-agnostic pretraining for VLAs. In *ICML*, 2026.

Andreas Steiner, André Susano Pinto, Michael Tschannen, Daniel Keysers, Xiao Wang, Yonatan Bitton, Alexey Gritsenko, Matthias Minderer, Anthony Sherbondy, Shangbang Long, et al. PaliGemma 2: A family of versatile VLMs for transfer. *arXiv preprint arXiv:2412.03555*, 2024.

Qi Sun, Pengfei Hong, Pala Tej Deep, Vernon Toh, U-Xuan Tan, Deepanway Ghosal, and Soujanya Poria. Emma-X: An embodied multimodal action model with grounded chain of thought and look-ahead spatial reasoning. In *ACL*, 2025.

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 *ICLR*, 2025.

Homer Rich Walke, Kevin Black, Tony Z. Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. BridgeData V2: A dataset for robot learning at scale. In *CoRL*, 2023.

Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, Siteng Huang, Yifan Tang, Wenhui Wang, Ru Zhang, Jianyi Liu, and Donglin Wang. VLA-Adapter: An effective paradigm for tiny-scale vision-language-action model. In *AAAI*, 2026a.

Yuqi Wang, Xinghang Li, Wenxuan Wang, Junbo Zhang, Yingyan Li, Yuntao Chen, Xinlong Wang, and Zhaoxiang Zhang. Unified vision-language-action model. In *ICLR*, 2026b.

Junjie Wen, Yichen Zhu, Minjie Zhu, Zhibin Tang, Jinming Li, Zhongyi Zhou, Xiaoyu Liu, Chaomin Shen, Yaxin Peng, and Feifei Feng. DiffusionVLA: Scaling robot foundation models via unified diffusion and autoregression. In *ICML*, 2025.

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 *ICLR*, 2024.

Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. In *CVPR*, 2024.

Youguang Xing, Xu Luo, Junlin Xie, Lianli Gao, Heng Tao Shen, and Jingkuan Song. Shortcut learning in generalist robot policies: The role of dataset diversity and fragmentation. In *CoRL*, 2025.

Jiange Yang, Yansong Shi, Haoyi Zhu, Mingyu Liu, Kaijing Ma, Yating Wang, Gangshan Wu, Tong He, and Limin Wang. Como: Learning continuous latent motion from internet videos for scalable robot learning. In *CVPR*, 2026a.

Jianwei Yang, Reuben Tan, Qianhui Wu, Ruijie Zheng, Baolin Peng, Yongyuan Liang, Yu Gu, Mu Cai, Seonghyeon Ye, Joel Jang, Yuquan Deng, and Jianfeng Gao. Magma: A foundation model for multimodal AI agents. In *CVPR*, 2025.

Shuai Yang, Hao Li, Bin Wang, Yilun Chen, Yang Tian, Tai Wang, Hanqing Wang, Feng Zhao, Yiyi Liao, and Jiangmiao Pang. Vision-language-action instruction tuning: From understanding to manipulation. In *ICLR*, 2026b.

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 *ICLR*, 2025.

Yifu Yuan, Haiqin Cui, Yaoting Huang, Yibin Chen, Fei Ni, Zibin Dong, Pengyi Li, Yan Zheng, Hongyao Tang, and Jianye Hao. Embodied-r1: Reinforced embodied reasoning for general robotic manipulation. In *ICLR*, 2026.

Michal Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In *CoRL*, 2024.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

Jianke Zhang, Yanjiang Guo, Yucheng Hu, Xiaoyu Chen, Xiang Zhu, and Jianyu Chen. UP-VLA: A unified understanding and prediction model for embodied agent. In *ICML*, 2025a.

Jianke Zhang, Xiaoyu Chen, Yanjiang Guo, Yucheng Hu, and Jianyu Chen. VLM4VLA: Revisiting vision-language-models in vision-language-action models. In *ICLR*, 2026a.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, XinQiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, He Wang, Zhizheng Zhang, Li Yi, Wenjun Zeng, and Xin Jin. DreamVLA: A vision-language-action model dreamed with comprehensive world knowledge. In *NeurIPS*, 2025b.

Wenyao Zhang, Bozhou Zhang, Zekun Qi, Wenjun Zeng, Xin Jin, and Li Zhang. Disentangled robot learning via separate forward and inverse dynamics pretraining. In *ICLR*, 2026b.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, Ankur Handa, Tsung-Yi Lin, Gordon Wetzstein, Ming-Yu Liu, and Donglai Xiang. CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. In *CVPR*, 2025.

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 T. 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 *CoRL*, 2023.

A Implementation Details

A.1 Inverse Dynamics Head

We implement the inverse dynamics head as a lightweight two-view MLP decoder. Let Z^{cur} and Z^{fut} denote the encoded visual token features of the current and future observations, respectively:

$$Z^{\mathrm{cur}}, Z^{\mathrm{fut}} \in \mathbb{R}^{B\times P\times C},$$

where B is the batch size, P is the number of visual tokens produced by the vision encoder, and C is the token feature dimension. We first compute their token-wise difference:

$$Z^{\mathrm{diff}} = Z^{\mathrm{fut}} - Z^{\mathrm{cur}}.$$

We then concatenate the current, future, and difference features along the channel dimension:

$$Z^{\mathrm{cat}} = [Z^{\mathrm{cur}}, Z^{\mathrm{fut}}, Z^{\mathrm{diff}}] \in \mathbb{R}^{B\times P\times 3C}.$$

Each concatenated token is processed independently by a patch-wise fusion MLP, which projects the feature dimension from $3C$ to a decoder dimension. The resulting fused token features are flattened across the token dimension and passed to an action MLP, which predicts an action chunk:

$$\check{a}_{t:t+H} = h_\psi(Z^{\mathrm{cur}}, Z^{\mathrm{fut}}) \in \mathbb{R}^{B\times H\times d_a},$$

where H denotes the action chunk length used by the corresponding VLA model and $d_a = 7$ denotes the action dimension.

For models with multiple visual inputs or camera views, we apply the inverse dynamics head independently to each view. For each view, we encode the observations at t and $t + H$, compute the corresponding inverse dynamics prediction, and evaluate the auxiliary loss using the same action target. The final inverse dynamics loss is averaged across all available views. The batch size, action chunk length, and other model-specific hyperparameters are reported in the corresponding implementation sections. The inverse dynamics head is used only during training and removed at inference time.

A.2 Pseudo Time Reversal

For auxiliary inverse dynamics learning, we additionally apply Pseudo Time Reversal (PTR) as a stochastic data augmentation. Given an inverse dynamics sample $(o_t, o_{t+H}, a_{t:t+H})$, PTR constructs a pseudo-reversed sample by swapping the observation order and reversing the action sequence after negating its motion offsets. For each action a_i, we define

$$a_i^\dagger = [-\Delta x_i, -\Delta y_i, -\Delta z_i, -\Delta\phi_i, -\Delta\theta_i, -\Delta\psi_i, g_i],$$

where the six motion offsets are negated while the gripper command g_i is kept unchanged. The pseudo-reversed counterpart of the action chunk $a_{t:t+H}$ is then defined as

$$a^\dagger_{t+H:t} = (a^\dagger_{t+H-1}, a^\dagger_{t+H-2}, \ldots, a^\dagger_t).$$

PTR is applied to each auxiliary sample with probability p_{rev}. The original sample $(o_t, o_{t+H}, a_{t:t+H})$ is retained with probability $1 - p_{\mathrm{rev}}$, while with probability p_{rev} it is replaced by

$$(o_{t+H}, o_t, a^\dagger_{t+H:t}).$$

PTR is used for all +Ours variants in our experiments, while the value of p_{rev} is selected separately for each model and reported in the corresponding implementation table. PTR is applied only to the auxiliary inverse dynamics objective and does not modify the action targets used by the main VLA policy.

We refer to this transformation as pseudo time reversal because negating the motion offsets yields motion in the opposite direction relative to the original trajectory, rather than the physically exact action sequence that would return the robot from o_{t+H} to o_t.

Table 7: Arm-action similarity among locally similar state pairs in the CALVIN ABC training data. Pairs are sampled within the same language-annotated trajectory segment using an end-effector position radius of 0.1. Cosine similarity is computed over the 6-DoF arm-action chunk, excluding the gripper command. The same 19,990 state pairs are used for both horizons.

Horizon	Median	$\cos > 0.7$	$\cos > 0.8$	$\cos > 0.9$
$H = 1$	0.708	50.6%	42.4%	29.3%
$H = 10$	0.631	44.5%	34.3%	18.4%

B Additional Analysis for Method Motivation

B.1 Action Similarity among Nearby States

To examine whether nearby robot states can receive similar action supervision under standard VLA training, we perform a data-level analysis on the CALVIN ABC training set. We represent each robot state using the 6-DoF end-effector pose, consisting of Cartesian position and Euler angles. State pairs are sampled within the same language-annotated trajectory segment using the `radius_random` procedure. Specifically, for each sampled anchor frame, candidate frames are restricted to those whose end-effector positions lie within an Euclidean distance of 0.1, after which a neighbor is sampled at random without replacement. Self-pairs are excluded, and duplicate unordered pairs are removed. We additionally require both endpoints to contain complete action chunks up to $H = 10$. This procedure yields 19,990 state pairs, which are shared across the $H = 1$ and $H = 10$ analyses.

For each pair (i, j), we compare the corresponding arm-action chunks using cosine similarity. Given an H-step chunk, we concatenate the first six action dimensions across time,

$$v_i^{(H)} = \mathrm{vec}\left(a_{i:i+H,:6}\right) \in \mathbb{R}^{6H},$$

and compute

$$\mathrm{sim}_H(i, j) = \frac{\left(v_i^{(H)}\right)^\top v_j^{(H)}}{\left\|v_i^{(H)}\right\|_2 \left\|v_j^{(H)}\right\|_2}.$$

Thus, the $H = 1$ analysis compares 6-dimensional arm actions, whereas the $H = 10$ analysis compares the full 60-dimensional action chunks. The gripper command is excluded. Because cosine similarity normalizes each complete action vector by its L2 norm, the comparison is insensitive to the overall magnitude of the chunk while retaining its directional and temporal structure.

As shown in Table 7, many locally similar states receive highly similar action supervision. At $H = 1$, the median cosine similarity is 0.708, with 42.4% of pairs exceeding 0.8 and 29.3% exceeding 0.9. The same tendency remains over the longer $H = 10$ horizon, where the median similarity is 0.631 and 34.3% of pairs exceed 0.8.

This analysis does not imply that similar action targets necessarily cause state aliasing. Rather, it shows that fine-grained differences between nearby states are not always accompanied by strongly differentiated action supervision. Standard action prediction objectives may therefore provide limited pressure to preserve some of these distinctions in the visual representation, motivating additional supervision that explicitly encourages fine-grained state discrimination.

C Diagnostic Analysis Details

C.1 Evaluation Data and Visual Encoders

We conduct the diagnostic analysis on the 40 tasks from the four LIBERO downstream suites: Spatial, Object, Goal, and Long. For both the state-difference and action probes, we use only the training demonstrations and take the agent-view RGB observation as visual input. All visual inputs are provided at a fixed resolution of 224×224, with model-specific preprocessing applied using each encoder's native image processor. The visual encoder is kept frozen throughout probe training.

We evaluate seven pretrained visual encoders: CLIP ViT-L/14, DINOv2 ViT-L, Florence-2-base, PaliGemma 2 3B, Qwen3-VL-8B-Instruct, SigLIP SO400M/14, and V-JEPA 2. For each model, we extract token-level visual features immediately before the model-specific projection layer when such a layer is present. We retain these features at the token level rather than pooling them into a single image representation. The probe-specific processing of the extracted visual tokens is described in the following sections.

C.2 State-Difference Probe

We construct temporally nearby observation pairs within each LIBERO training demonstration. For an episode of length L, we use every eligible anchor frame $i = 0, \ldots, L - 11$ and sample a single temporal offset

$$\delta_i \sim \mathrm{Uniform}\{1, \ldots, 10\}, \qquad j = i + \delta_i.$$

Thus, each anchor is paired with one future observation within 1–10 stored frames. The sampled pair is fixed throughout probe training, and the same pairs are used for all visual encoders. We use a pair sampling seed of 123, resulting in 233,077 fixed observation pairs across the 40 LIBERO tasks.

The robot state associated with each observation is represented as

$$s = [x, y, z, r_x, r_y, r_z, f_0, f_1] \in \mathbb{R}^8,$$

where (x, y, z) denotes the Cartesian end-effector position, (r_x, r_y, r_z) is the axis-angle rotation vector, and (f_0, f_1) are the two finger coordinates. For a pair (i, j), we define the state-difference target by coordinate-wise subtraction,

$$\Delta s_{ij} = s_j - s_i.$$

This subtraction is applied to all eight state dimensions, including the rotation-vector and finger coordinates.

To account for differences in scale across state dimensions, we normalize each target dimension using the population standard deviation computed over all 233,077 fixed training pairs pooled across the 40 tasks:

$$y_{ij,d} = \frac{\Delta s_{ij,d}}{\sigma_d}.$$

We do not subtract the mean or clip the normalized targets. If $\sigma_d \leq 10^{-6}$, its normalization scale is set to 1.

For each observation pair, the frozen visual encoder produces token-level features

$$Z_i = f(o_i), \qquad Z_j = f(o_j) \in \mathbb{R}^{P \times D},$$

where P and D depend on the encoder. We compute the token-wise feature difference

$$\Delta Z_{ij} = Z_j - Z_i$$

by subtracting corresponding visual tokens before any model-specific projection layer. We do not apply mean pooling or provide language, action, temporal-gap, or robot-state information as additional probe inputs.

The probe g_θ operates directly on ΔZ_{ij}. A shared token-wise linear layer first projects each D-dimensional token to 256 dimensions, followed by a GELU activation. The projected token features are then flattened into a $P \cdot 256$-dimensional vector and passed through an MLP with hidden dimensions 1024 and 512, followed by an 8-dimensional output layer. GELU activations are applied between the hidden layers. We train a separate probe for each visual encoder, while a single probe covers all 40 LIBERO tasks for that encoder. Because the number and dimensionality of visual tokens vary across encoders, the probe follows the same architecture but does not have an identical parameter count across models.

We train the probe using mean squared error on the normalized state-difference target:

$$\mathcal{L}_{\mathrm{state}} = \frac{1}{8B} \sum_{b=1}^{B} \| g_\theta(\Delta Z_b) - y_b \|_2^2 .$$

All probes are optimized with AdamW for 5,000 steps using a batch size of 64 and a learning rate of 2×10^{-4}. During training, tasks are sampled uniformly from the 40 LIBERO tasks, and pairs are drawn from the fixed pair set of the selected task.

After training, we freeze the final probe and evaluate it once over the complete set of 233,077 fixed training pairs. We report the pooled normalized MSE,

$$\mathrm{MSE}_{\mathrm{pooled}} = \frac{1}{8N} \sum_{(i,j)\in\mathcal{P}} \|g_\theta(\Delta Z_{ij}) - y_{ij}\|_2^2, \qquad N = 233{,}077.$$

This is a training-pair reconstruction metric rather than a held-out generalization metric, and each pair receives equal weight in the reported value. Lower loss indicates that fine-grained robot-state differences are better preserved in the frozen visual representation under the same probing protocol.

C.3 Action Probe

For the action probe, we freeze the visual encoder and train only a lightweight MLP policy head. The probe takes visual features from a single current observation without language or task information. Since task identity is not provided as input, we train a separate probe for each of the 40 tasks across LIBERO-Spatial, Object, Goal, and Long.

Given frozen visual tokens

$$Z \in \mathbb{R}^{B\times P\times D},$$

each token is first projected independently to 256 dimensions using a shared linear layer followed by GELU. The projected tokens are then flattened and passed through an MLP with hidden dimensions 1024 and 512. The output layer predicts a four-step action chunk,

$$\hat{a}_{t:t+4} \in \mathbb{R}^{B\times 4\times 7},$$

where each action consists of six arm-motion dimensions and one gripper dimension. The same probe architecture is used for all encoders, with the input dimensions adjusted according to the number and dimensionality of their visual tokens.

The six arm-action dimensions are normalized using task-specific action statistics and the fixed action range used by the LIBERO data loader. The gripper target is treated as a binary variable, and the probe predicts its logit directly. Invalid or padded steps in the four-step action chunk are excluded from the loss.

We train the probe using an ℓ_1 loss for the normalized arm actions and binary cross-entropy for the gripper prediction:

$$\mathcal{L}_{\mathrm{action}} = \mathcal{L}_{\mathrm{arm}} + \lambda_g \mathcal{L}_{\mathrm{gripper}}, \qquad \lambda_g = 0.01.$$

Each task-specific probe is optimized with AdamW for 1,000 steps using a batch size of 64 and a learning rate of 2×10^{-4}.

For each task, we record the training loss averaged over the final training epoch segment. We report the action probe loss as the unweighted mean of these values across the 40 tasks. Lower loss indicates that the frozen visual representation supports a more accurate mapping from the current observation to the corresponding low-level actions.

C.4 Action Probe Rollout Evaluation

We directly deploy the same task-specific action probes as policies without additional training. For each LIBERO task, we evaluate 20 rollouts using the first 20 predefined initial states provided by the benchmark. Each episode begins with 10 no-op settling steps, followed by at most 220, 280, 300, and 520 policy-controlled environment steps for LIBERO-Spatial, Object, Goal, and Long, respectively, with early termination upon success.

At each policy query, the probe predicts a four-step action chunk, and all four actions are executed sequentially before obtaining a new prediction. Predicted actions are converted back to the LIBERO action space by reversing the action preprocessing used during probe training.

We report task success rates over 20 trials, corresponding to 200 rollouts per suite and 800 rollouts per visual encoder in total.

Table 8: Action probe rollout success rates across the four LIBERO downstream suites. Each suite contains 10 tasks with 20 rollouts per task. The average is computed across all 40 tasks.

Visual Encoder	Spatial	Object	Goal	Long	Avg.
CLIP	74.0	81.0	61.5	23.5	60.0
DINOv2	66.5	72.0	69.5	42.5	62.6
PaliGemma2	70.5	74.5	66.5	27.5	59.8
SigLIP	78.0	78.5	64.0	27.5	62.0
V-JEPA 2	76.0	72.0	60.0	28.0	59.0
Qwen3-VL	49.5	39.5	43.0	9.5	35.4
Florence-2	81.5	71.5	56.0	16.0	56.3

C.5 Action Probe Rollout Results by LIBERO Suite

Table 8 reports the rollout success rates of the task-specific action probes separately for the four LIBERO downstream suites. Each value is averaged over the 10 tasks within the corresponding suite, with 20 rollouts per task. The overall average is computed across all 40 tasks. Because the visual feature dimensionality differs across encoders, the resulting action probe MLPs also differ in parameter count; therefore, these rollout results should be interpreted as a complementary evaluation rather than a capacity-controlled comparison across encoders.

D LIBERO Experimental Details

D.1 LIBERO-90 Pretraining

For LIBERO-90 pretraining, we use VLM4VLA with Qwen3-VL-8B-Instruct as the backbone and train for one epoch using the static RGB view. We use a batch size of 256, a learning rate of 2×10^{-5}, and a warmup of 0.25 epoch. The model predicts four-step, 7-dimensional action chunks. The vanilla and +Ours models use the same base training configuration, while +Ours additionally applies auxiliary inverse dynamics with $\lambda_{\text{inv}} = 0.1$ and PTR with $p_{\text{rev}} = 0.5$. The inverse dynamics decoder dimension is 1024. The resulting vanilla and +Ours checkpoints are used to initialize their corresponding downstream fine-tuning runs.

D.2 Downstream Fine-Tuning

For downstream evaluation, we jointly fine-tune VLM4VLA on the training data from all 40 tasks across LIBERO-Spatial, Object, Goal, and Long. The four suite datasets are sampled in proportion to their numbers of transitions, without task-uniform reweighting. The vanilla model is initialized from the vanilla LIBERO-90 pretrained checkpoint, while +Ours is initialized from the corresponding LIBERO-90 checkpoint trained with auxiliary inverse dynamics and PTR.

We fine-tune the full VLA model, including the vision encoder, for one epoch. The model takes a 224×224 agent-view RGB observation together with the language instruction and predicts a four-step, 7-dimensional action chunk. We use a learning rate of 2×10^{-5}, an effective global batch size of 256, and no learning-rate warmup. For +Ours, auxiliary inverse dynamics and PTR remain active during downstream fine-tuning using the same auxiliary configuration as in LIBERO-90 pretraining.

We evaluate all 40 downstream tasks with 20 rollouts per task, corresponding to 200 rollouts per suite and 800 rollouts in total. We use the predefined LIBERO initial states and allow at most 220, 280, 300, and 520 policy-controlled environment steps for Spatial, Object, Goal, and Long, respectively, with early termination upon success. We report the mean success rate for each suite and the average across all 40 tasks.

E CALVIN Experimental Details

We follow the standard CALVIN ABC→D protocol, training on demonstrations from environments A, B, and C and evaluating in the unseen D environment. Each model is evaluated over 1,000 rollout

sequences, and we report success rates for completing 1–5 consecutive subtasks together with the average completed sequence length. We retain the original training pipeline of each baseline and add auxiliary inverse dynamics and PTR for the +Ours variant. Model-specific configurations are described below.

E.1 VLM4VLA

For VLM4VLA, we follow the released CALVIN configuration and evaluate both Qwen3-VL-8B-Instruct and Qwen3-VL-2B-Instruct backbones using the static RGB view as input. Both variants are trained for 30K steps with a batch size of 128 and a learning rate of 2×10^{-5}, and predict 10-step, 7-dimensional action chunks. The 8B variant uses 0.25 epochs of warmup, while the 2B variant is trained without warmup.

For +Ours, we use $\lambda_{\text{inv}} = 0.1$ and $p_{\text{rev}} = 0.5$ for the 8B variant, and $\lambda_{\text{inv}} = 0.02$ and $p_{\text{rev}} = 0.3$ for the 2B variant. Both use an inverse dynamics decoder dimension of 1024. All variants are evaluated at the 30K checkpoint.

E.2 FLOWER

For FLOWER, we follow the default configuration released with the official implementation and initialize the model from the released FLOWER pretrained checkpoint. The model uses both static and wrist RGB views and is trained for 40K steps with a batch size of 32 and a learning rate of 2×10^{-5}. We retain its original three-phase learning-rate schedule and 10-step, 7-dimensional action prediction. For +Ours, we use $\lambda_{\text{inv}} = 0.3$, $p_{\text{rev}} = 0.5$, and an inverse dynamics decoder dimension of 1024.

Our reproduced vanilla FLOWER model did not reach performance comparable to the published result, so the vanilla FLOWER result reported in the main paper is taken from the original work. The +Ours variant is trained with the released configuration augmented with auxiliary inverse dynamics and PTR and is evaluated using the final checkpoint.

E.3 GR-1

For GR-1, since the official release does not provide training code, we base our implementation on the publicly available GR1-Training repository.[1] We initialize both the vanilla and +Ours variants from the released GR-1 pretrained checkpoint and retain its original action and future-frame prediction objectives. The model uses both static and wrist RGB observations at 224×224 resolution and is trained for 20 epochs with a maximum learning rate of 9×10^{-4} and one warmup epoch. It predicts 10-step, 7-dimensional action chunks and uses a temporal skip of three frames for future prediction.

For +Ours, the inverse dynamics branch predicts three 7-dimensional actions and uses an auxiliary loss weight of 10, a PTR probability of 0.5, and a decoder dimension of 1024. We use a maximum gradient norm of 2.0 for +Ours, compared with 1.0 for the vanilla GR-1 baseline. Both variants are evaluated using their 20-epoch checkpoints.

E.4 Ablation on Pseudo Time Reversal

We ablate PTR using VLM4VLA 8B on CALVIN ABC→D. As shown in Table 9, auxiliary inverse dynamics without PTR already improves the average completed sequence length from 4.03 to 4.09. Adding PTR further increases it to 4.10, with small additional gains on most sequence lengths. These results indicate that the main improvement comes from auxiliary inverse dynamics, while PTR provides a modest complementary benefit.

[1] https://github.com/EDiRobotics/GR1-Training

Table 9: Ablation of Pseudo Time Reversal on CALVIN ABC→D using VLM4VLA. We compare vanilla VLA training, auxiliary inverse dynamics without PTR, and the full +Ours variant with PTR. We report success rates over 1000 rollouts for completing 1–5 tasks in a row and the average completed sequence length. Bold numbers indicate the best results in each column.

Method	1	2	3	4	5	Avg. Len. ↑
VLM4VLA 8B (Zhang et al., 2026a)	93.4	85.9	79.7	75.2	69.2	4.03
+ Inverse Dynamics	94.3	**87.1**	81.1	76.3	70.2	4.09
+ **Ours (with PTR)**	**94.4**	86.7	**81.5**	**76.6**	**70.5**	**4.10**

F SimplerEnv Experimental Details

F.1 SpatialVLA on Bridge

We use SpatialVLA as the baseline for SimplerEnv. We initialize from the released SpatialVLA pretrained checkpoint, fine-tune the model on Bridge, and evaluate it on four WidowX manipulation tasks: putting a spoon on a towel, putting a carrot on a plate, stacking a green block on a yellow block, and putting an eggplant into a yellow basket. For each task, we run 24 trials with randomized initializations and report the final task success rate.

Except for the auxiliary inverse dynamics objective and PTR used in our +Ours variant, we keep the training configuration identical to the official implementation. Since the official release does not specify the exact fine-tuning duration or checkpoint selection protocol, we use a fixed training budget of 30K steps for both variants and evaluate the final checkpoint. We use a batch size of 256 and a learning rate of 2×10^{-5}. The model predicts four-step, 7-dimensional action chunks from the static RGB view. For +Ours, we use $\lambda_{\text{inv}} = 0.2$, $p_{\text{rev}} = 0.5$, and an inverse dynamics decoder dimension of 512.

For comparison, the OpenVLA and RoboVLMs results in Table 5 are taken from (Qu et al., 2025). We additionally re-evaluate the publicly released SpatialVLA pretrained checkpoint under the same SimplerEnv evaluation protocol.

G State-Feature Alignment Analysis

We analyze state-feature alignment on CALVIN by measuring whether changes in end-effector pose are reflected in the visual feature space beyond low-level image differences. The analysis uses language-annotated CALVIN validation trajectories. We first collect all frames that fall within language-annotation intervals, resulting in 64,793 annotated validation records. Frame pairs are sampled only within the same language-annotated trajectory segment, so that the two frames share the same instruction and high-level task context while the robot state varies. We consider temporal gaps of $\{1, 2, 4, 8, 16, 32\}$ and uniformly sample 1,000 pairs for each gap, yielding 6,000 frame pairs in total.

For each frame i, we use the 6-DoF end-effector pose s_i, corresponding to the first six dimensions of `robot_obs`. Let $\sigma \in \mathbb{R}^6$ denote the per-dimension standard deviation of these pose dimensions over all annotated validation records. We define the normalized pose distance between frames i and j as

$$d_{\text{pose}}(i, j) = \left\| \frac{s_i - s_j}{\sigma} \right\|_2 . \tag{8}$$

For visual features, we extract spatial visual tokens from the selected vision encoder layer. The main analysis uses layer 24. Let h_i^k and h_j^k denote the visual tokens at spatial position k, and let

$$\bar{h}_i = \frac{1}{K} \sum_{k=1}^{K} h_i^k$$

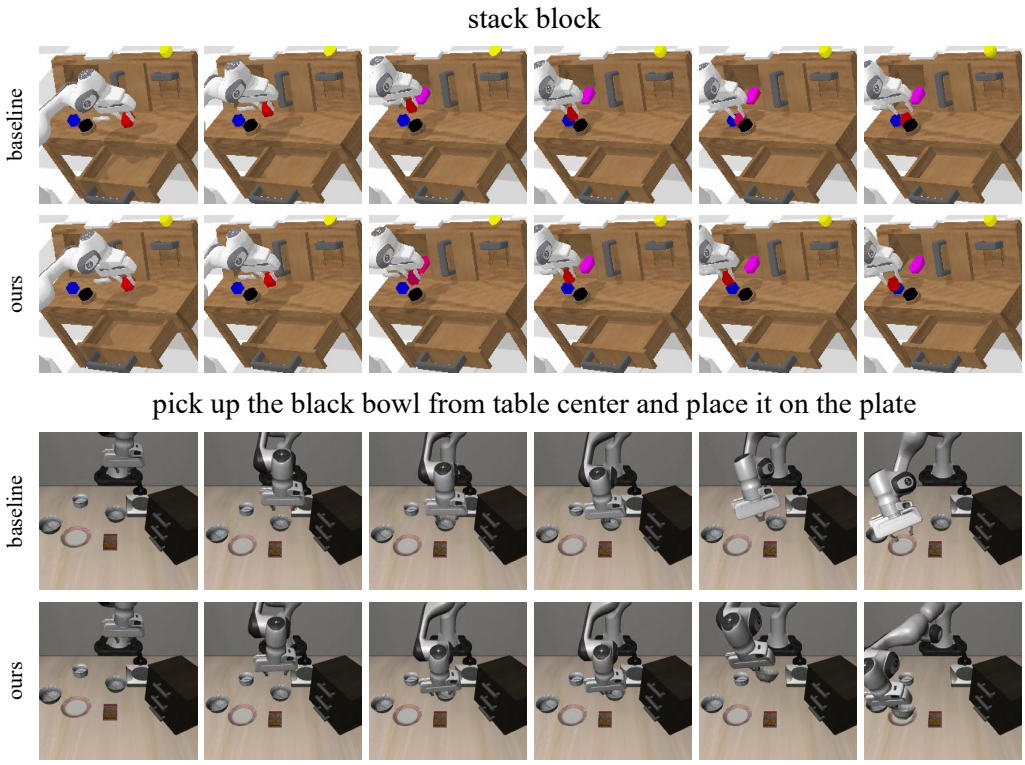


Figure 3: **Qualitative results on CALVIN ABC→D and LIBERO.** Each rollout is shown with six representative frames. **Top:** In the CALVIN stack-block task, the vanilla VLM4VLA 8B policy releases the red block before it is sufficiently aligned above the blue block, whereas +Ours successfully places the red block on the target. **Bottom:** In the LIBERO-Object task, the action-probe policy trained on features from the vanilla VLA encoder fails to grasp the black bowl, whereas the probe trained on features from +Ours approaches the bowl more accurately, grasps it, and moves it toward the plate.

denote the mean-pooled visual feature. We consider two complementary feature distances. The cosine distance measures changes in feature direction,

$$d_{\cos}(i,j) = 1 - \frac{\langle \bar{h}_i, \bar{h}_j \rangle}{\|\bar{h}_i\|_2 \|\bar{h}_j\|_2}, \tag{9}$$

while the scale difference measures changes in feature magnitude,

$$d_{\text{scale}}(i,j) = \left| \|\bar{h}_i\|_2 - \|\bar{h}_j\|_2 \right|. \tag{10}$$

To account for low-level visual changes, we compute a pixel-controlled partial Spearman correlation. For each frame pair, d_{pix} is measured by pixel MSE between downsampled grayscale image thumbnails. We first rank-transform the feature distance, pose distance, and pixel distance. The ranked feature and pose distances are then separately residualized with respect to the ranked pixel distance, and the final score is computed as the Pearson correlation between the two residual vectors:

$$\rho_{\text{partial}} = \text{corr}\left(\text{resid}\big(\text{rank}(d_{\text{feat}}), \text{rank}(d_{\text{pix}})\big), \text{resid}\big(\text{rank}(d_{\text{pose}}), \text{rank}(d_{\text{pix}})\big)\right). \tag{11}$$

Higher partial correlation indicates that changes in end-effector pose are more strongly reflected in the learned visual representation after accounting for low-level pixel differences.

H Additional Qualitative Results

Figure 3 provides qualitative examples from CALVIN and LIBERO. On the CALVIN stack-block task, the vanilla VLM4VLA 8B policy releases the red block before it is sufficiently aligned with

the blue block, whereas +Ours maintains a more precise approach and successfully completes the stacking motion. On LIBERO-Object, the action probe policy trained on features from the vanilla VLA encoder fails to establish a successful grasp of the black bowl, while the probe trained on features from +Ours approaches the bowl more accurately, grasps it, and proceeds toward the target plate. These examples qualitatively illustrate the improvements in precise action prediction observed in our quantitative evaluations.