VIDAL: A VISUAL DYNAMICS-GROUNDED ACTION LATENT SPACE FOR VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language-Action (VLA) models have become a central paradigm for robot policy learning and predict actions in three forms: raw action chunks, discrete action tokens, or continuous action latents. However, existing action representations primarily model action trajectories, with limited consideration of the visual dynamics induced by these actions. We introduce ViDAL, a Visual Dynamics-grounded Action Latent Space that anchors continuous action latents in the future visual dynamics of the scene. Specifically, ViDAL learns an action latent space by training an Action Variational Autoencoder (Action VAE) to reconstruct action chunks while aligning its latent with future scene dynamics through a predictor conditioned on the current visual state. When integrated into downstream robot policies, the proposed Action VAE serves as a plug-in action interface compatible with multiple VLA architectures and enables optional future-video prediction as an additional capability. Empirically, ViDAL outperforms competitive baselines on LIBERO with 98.1% average success, improves a multi-task $\pi_{0.5}$ policy on RoboTwin 2.0 from 54.3% to 65.5% (Clean) and from 33.2% to 43.1% (Random) success rates over 50 dual-arm tasks, and yields 20.0% and 23.4% absolute success-rate gains on real-world single-arm Franka and dual-arm ARX robot platforms.

1 INTRODUCTION

Vision-Language-Action (VLA) policies map visual observations and language instructions to executable robot actions (Brohan et al., 2022; Zitkovich et al., 2023; Kim et al., 2024; Black et al., 2024; Intelligence et al., 2025; Liu et al., 2024). Their action representation defines both what the policy learns to predict and what the robot executes. Robot datasets increasingly span different tasks, scenes, and embodiments (Walke et al., 2023; O'Neill et al., 2024; Khazatsky et al., 2024). For a VLA trained on such data, the action target must represent executable motion and the scene changes that make that motion useful, from establishing contact to moving an object.

Current VLAs predict raw action chunks (Black et al., 2024; Intelligence et al., 2025; Chi et al., 2025), discrete action tokens (Pertsch et al., 2025; Wang et al., 2025; Zhou et al., 2025; Liu et al., 2025), or continuous action latents (Tan et al., 2024). Action tokenizers and autoencoders learn compact codes by reconstructing action chunks (Pertsch et al., 2025; Wang et al., 2025; Zhou et al., 2025; Liu et al., 2025; Tan et al., 2024), as illustrated in Fig. 1(a). Reconstruction preserves the commands, but does not directly distinguish motion by its effect on the scene. The latent also defines the target learned by the downstream policy, so its structure determines which differences between action chunks the policy must resolve. Meanwhile, VLAs with future prediction use video or world-model signals to guide policy learning (Assran et al., 2025; Hu et al., 2024; Cen et al., 2025; Xu et al., 2026; Shang et al., 2026; Su et al., 2026), as illustrated in Fig. 1(b). Together, these lines of work motivate using future visual change to shape an executable action latent before VLA policy training.

The visual effect of an action depends on the scene in which it is executed. The same gripper displacement can move an object after contact or leave the scene unchanged when the gripper moves through free space. Near contact, a small change in timing or direction can alter the object's trajectory even when the action chunks look similar. The initial observation provides scene context, while the future observation reveals the resulting change in object pose and contact. Our insight is to make

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.

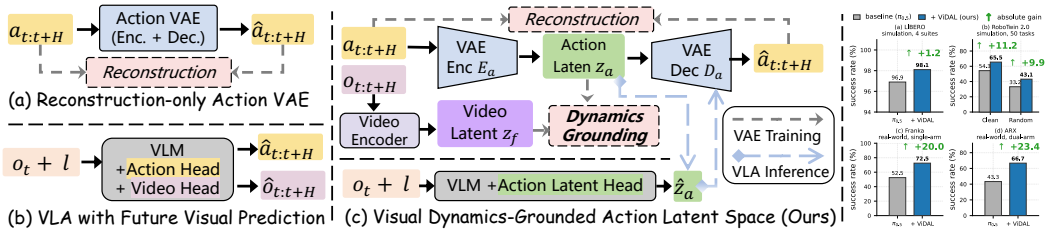


Figure 1: **Overview of ViDAL.** (a) Action VAE supervised only by reconstruction. (b) VLAs supervised with visual prediction signals. (c) ViDAL (ours) grounds the action latent space in future visual dynamics, and uses the Action VAE as a plug-in action representation at deployment. Right column: consistent success-rate gains on simulation benchmarks and real-robot platforms.

that change a training signal for the action latent: given the initial visual state, the latent should help predict the future visual state. Conditioning on the initial state assigns scene context to vision and controllable motion to the action code. Future visual prediction then encourages the code to retain motion that matters for manipulation, while action reconstruction keeps it tied to executable commands.

We propose ViDAL (Visual Dynamics-grounded Action Latent Space, Fig. 1(c)) to learn an executable action target from action chunks and their future visual consequences. A frozen video VAE encodes the initial and future observations, while an Action VAE encodes and reconstructs the aligned action chunk. A lightweight predictor uses the initial visual latent and the action latent to predict the future visual latent, training the action code to explain the scene transition. For downstream policy learning, the frozen Action VAE provides latent targets and decodes VLA predictions into executable actions. The predicted visual latents can also be decoded into future video to inspect the learned dynamics. This decoupled interface supports regression experts such as OpenVLA-OFT (Kim et al., 2025) and flow-matching experts such as $\pi_{0.5}$ (Intelligence et al., 2025) while retaining their VLM backbones.

We evaluate ViDAL with regression and flow-matching policies on LIBERO (Liu et al., 2023), RoboTwin 2.0 (Chen et al., 2025a), and physical Franka and ARX robots. On RoboTwin 2.0, it improves a multi-task $\pi_{0.5}$ policy by 11.2 points on Clean and 9.9 points on Random over 50 dual-arm tasks. The gains on Franka and ARX are 20.0 and 23.4 points. On LIBERO, it reaches 98.1% average success, the highest among the compared methods. Removing future visual supervision reduces RoboTwin success on both splits, linking the representation objective to downstream control. Together, these results support visual-dynamics grounding as a reusable principle across VLA action experts and robot embodiments.

Our contributions are summarized as follows:

- We introduce ViDAL, which, to our knowledge, is the first approach to ground action latents in future visual dynamics for VLA models, learned via an Action VAE jointly trained on action reconstruction and future visual latent prediction.
- We propose a plug-in action-latent interface for downstream VLA policies, where the frozen Action VAE serves as the action representation for both regression and flow-matching action experts, requiring no changes to the VLM backbone.
- We conduct extensive experiments in both simulation (LIBERO, RoboTwin 2.0) and real-world (single-arm Franka, dual-arm ARX) environments. Results show that ViDAL consistently improves both regression and flow-matching VLA policies and surpasses competitive action-representation methods.

2 Related Work

Vision-Language-Action Models. VLA models adapt pretrained vision-language backbones to predict robot actions from observations and instructions (Brohan et al., 2022; Zitkovich et al., 2023; Kim et al., 2024; Driess et al., 2023; Li et al., 2023; Team et al., 2024; Liu et al., 2024). Flow-matching experts in π_0 (Black et al., 2024) and $\pi_{0.5}$ (Intelligence et al., 2025) generate continuous

actions, while OpenVLA-OFT (Kim et al., 2025) and StarVLA-OFT (Ye et al., 2026a) use one-step regression experts. ViDAL changes their action target to a dynamics-grounded latent and decodes it to executable actions while retaining the VLA backbone and expert.

Action Representations for VLA Models. VLA action representations take three predominant forms, each shaped by a different supervision signal. Raw action chunks are regressed under behavior cloning (Black et al., 2024; Intelligence et al., 2025; Liu et al., 2024; Chi et al., 2025; Zhao et al., 2023). Discrete action tokens compress trajectories via quantization (Pertsch et al., 2025; Wang et al., 2025; Zhou et al., 2025; Liu et al., 2025), while continuous action latents use an autoencoder bottleneck (Tan et al., 2024). These targets are learned from action trajectories. ViDAL adds future visual change as direct supervision for the latent action representation.

Visual Dynamics for VLA Policies. Visual dynamics enters VLA policies through video-pretrained features (Assran et al., 2025; Sun et al., 2026), auxiliary future-frame or video prediction (Hu et al., 2024; Jia et al., 2025; Ye et al., 2026b), and world-model signals for action reasoning (Cen et al., 2025; Shang et al., 2026; Xu et al., 2026; Su et al., 2026). These methods use visual dynamics to shape the policy or its input features. ViDAL instead uses future visual change to shape the action latent before policy training, then freezes the Action VAE as a common target for different backbones and experts.

3 METHOD

3.1 OVERVIEW

ViDAL learns a reusable action code by coupling action reconstruction to conditional future visual prediction. Given an observation clip $\mathbf{o}_{t:t+H}$ and its aligned action chunk $\mathbf{a}_{t:t+H-1}$, a frozen video VAE extracts initial and future visual latents, while an Action VAE encodes and reconstructs the action chunk (Fig. 2). A lightweight predictor uses the initial visual latent and the action latent to predict the future visual latent. Its loss updates the Action VAE encoder, shaping the target later learned by the VLA (Section 3.2). The Action VAE is then frozen and used as the action target and decoder for VLA policy learning (Section 3.3).

3.2 LEARNING THE ACTION LATENT SPACE

The central challenge in learning action latents is that action reconstruction alone does not determine which aspects of an action chunk are relevant to policy learning: a low-dimensional latent may reconstruct actions while failing to encode how those actions interact with the current scene. ViDAL addresses this by adding visual-dynamics supervision, encouraging the action latent to be informative for predicting the future visual latent conditioned on the initial observation.

Action VAE. The Action VAE maps the action chunk to a continuous action latent $\mathbf{z}_a \in \mathbb{R}^{T\times D}$ via a Gaussian posterior $q_\phi(\mathbf{z}_a \mid \mathbf{a}_{t:t+H-1}) = \mathcal{N}(\boldsymbol{\mu}_\phi, \mathrm{diag}(\boldsymbol{\sigma}_\phi^2))$, where the encoder applies a $4\times$ temporal downsampling so that $T{=}H/4$ and D is the latent dimension. We sample $\mathbf{z}_a$ with the standard reparameterization trick and decode it back to the original action chunk:

$$\mathbf{z}_a = \boldsymbol{\mu}_\phi + \boldsymbol{\sigma}_\phi \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \qquad \hat{\mathbf{a}}_{t:t+H-1} = D_a^\psi(\mathbf{z}_a). \tag{1}$$

The continuous bottleneck preserves smooth trajectory structure and is compatible with both regression and flow-matching action experts.

Visual-Dynamics Supervision. A frozen video encoder E_v extracts an initial visual latent $\mathbf{z}_0$ from the current observation and a future visual latent $\mathbf{z}_f$ from the subsequent frames as the prediction target,

$$(\mathbf{z}_0, \mathbf{z}_f) = E_v(\mathbf{o}_{t:t+H}), \tag{2}$$

and a lightweight predictor P_θ regresses $\mathbf{z}_f$ from the initial visual latent and the action latent:

$$\hat{\mathbf{z}}_f = P_\theta(\mathbf{z}_0, \mathbf{z}_a). \tag{3}$$

Conditioning on $\mathbf{z}_0$ is essential because the same action chunk can correspond to different scene transitions in different contexts. The predictor therefore drives $\mathbf{z}_a$ to retain action-relevant scene-transition structure rather than only low-level action statistics. The frozen E_v provides a fixed visual

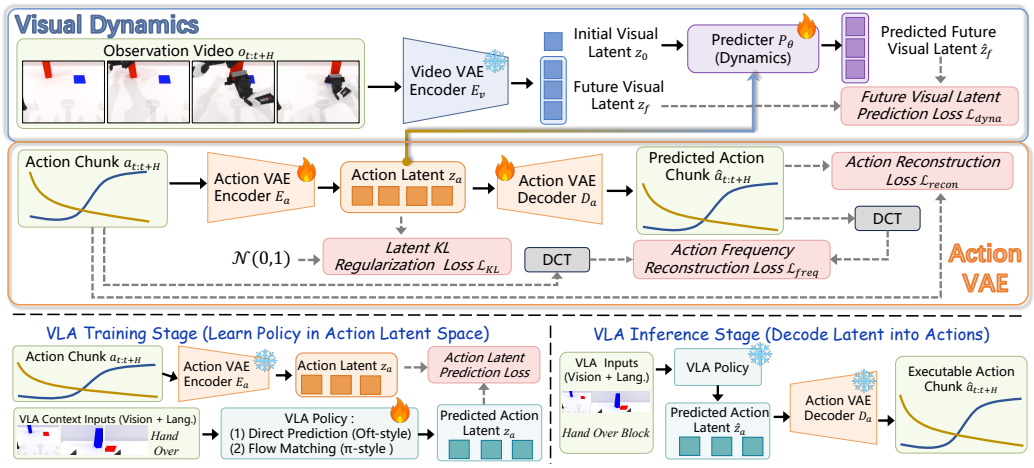


Figure 2: **Framework of ViDAL.** *Top:* the Action VAE is trained with visual-dynamics supervision from a frozen video encoder. *Bottom:* the frozen Action VAE then plugs into VLA training (left) and inference (right), serving as the action representation between the policy and the robot.

target, while the prediction loss updates the Action VAE encoder through P_θ. Both visual modules are discarded after representation learning, so deployment requires no video model.

Training Objective. The Action VAE is jointly trained with four complementary terms,

$$\mathcal{L}_{\text{ViDAL}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{freq}}\,\mathcal{L}_{\text{freq}} + \lambda_{\text{KL}}\,\mathcal{L}_{\text{KL}} + \lambda_{\text{dyna}}\,\mathcal{L}_{\text{dyna}}, \tag{4}$$

where $\mathcal{L}_{\text{rec}} = \|\hat{\mathbf{a}} - \mathbf{a}\|_1$ reconstructs the action chunk in the time domain, $\mathcal{L}_{\text{freq}} = \|\text{DCT}(\hat{\mathbf{a}}) - \text{DCT}(\mathbf{a})\|_1$ follows FAST's DCT-based action tokenization (Pertsch et al., 2025) to preserve temporal frequency structure, $\mathcal{L}_{\text{KL}} = D_{\text{KL}}(q_\phi(\mathbf{z}_a \,|\, \mathbf{a}) \,\|\, \mathcal{N}(\mathbf{0}, \mathbf{I}))$ regularizes the latent distribution, and $\mathcal{L}_{\text{dyna}} = \|\hat{\mathbf{z}}_f - \text{sg}(\mathbf{z}_f)\|_1$ aligns the action latent with future scene evolution. The stop-gradient operator $\text{sg}(\cdot)$ keeps the visual target space fixed.

3.3 Integrating ViDAL into VLA Policies

After representation learning, ViDAL replaces the raw-action interface of a VLA policy with an interface based on action latents: during training, the policy predicts an encoded action latent, and at inference, the predicted latent is decoded into an executable action chunk.

Plug-In Action Representation. The Action VAE is frozen and used as a plug-in action representation that the VLA policy is trained to predict. For each action chunk $\mathbf{a}_{t:t+H-1}$, the frozen encoder produces a deterministic latent target,

$$\bar{\mathbf{z}}_a = \text{sg}\big(\text{Norm}\big(\boldsymbol{\mu}_\phi(\mathbf{a}_{t:t+H-1})\big)\big)\,, \tag{5}$$

where $\text{Norm}(\cdot)$ is a fixed latent normalization estimated from the Action VAE training set, and the bar distinguishes this deterministic downstream target from the sampled latent $\mathbf{z}_a$ used in Action VAE training. This interface keeps the VLM backbone unchanged and only replaces the action-modeling target, so the same dynamics-grounded latent space can be plugged into different VLA architectures without modifying their backbones.

Regression Action Experts. For VLAs with regression action experts such as OpenVLA-OFT (Kim et al., 2025) and StarVLA-OFT (Ye et al., 2026a), the action expert maps the VLA context $\mathbf{c}$ (visual observation, language instruction, and policy features) to a predicted latent $\tilde{\mathbf{z}}_a$ and is trained against $\bar{\mathbf{z}}_a$:

$$\tilde{\mathbf{z}}_a = F_\omega(\mathbf{c}), \qquad \mathcal{L}_{\text{reg}} = \ell(\tilde{\mathbf{z}}_a, \bar{\mathbf{z}}_a)\,, \tag{6}$$

where $\ell(\cdot,\cdot)$ is the regression loss used by the action expert. The lightweight one-step expert is preserved while the supervised target moves from the raw action space to the dynamics-grounded latent space.

Flow-Matching Action Experts. For VLAs with flow-matching action experts such as π_0 and $\pi_{0.5}$ (Black et al., 2024; Intelligence et al., 2025), the same latent target serves as the endpoint of a conditional flow. With the interpolant $\bar{\mathbf{z}}_\tau = \tau\boldsymbol{\xi} + (1-\tau)\bar{\mathbf{z}}_a$ and noise $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, the action expert learns a vector field in the action latent space:

$$\mathcal{L}_{\text{flow}} = \mathbb{E}_{\tau,\boldsymbol{\xi},\bar{\mathbf{z}}_a}\left[\left\| v_\omega(\bar{\mathbf{z}}_\tau, \mathbf{c}, \tau) - (\boldsymbol{\xi} - \bar{\mathbf{z}}_a) \right\|_2^2 \right]. \tag{7}$$

The generative process of flow-matching policies is unchanged, and only the modeled distribution shifts from raw actions to dynamics-grounded action latents.

Inference. The VLA policy emits a normalized latent $\tilde{\mathbf{z}}_a$. We invert the latent normalization before decoding it into executable actions, $\hat{\mathbf{a}}_{t:t+H-1} = D_a^\psi(\text{Norm}^{-1}(\tilde{\mathbf{z}}_a))$. ViDAL therefore serves as an action representation layer between the VLA action expert and the robot, with no video model in the loop.

4 Experiments

We evaluate ViDAL through four questions. **Q1:** Does a dynamics-grounded action representation improve VLA success across simulation benchmarks and real robots (Sections 4.2 and 4.5)? **Q2:** Which training signals and policy targets produce the gain (Sections 4.3 and 4.4)? **Q3:** How does the gain change with the number of tasks and demonstrations (Section 4.4)? **Q4:** What visual information does the learned action latent retain (Section 4.6)?

4.1 Experimental Setup

Benchmarks. LIBERO (Liu et al., 2023) evaluates single-arm language-conditioned manipulation across four task suites. RoboTwin 2.0 (Chen et al., 2025a) evaluates 50 dual-arm tasks under Clean and randomized-scene Random conditions. We also evaluate on single-arm Franka and dual-arm ARX robots. We report task success rates (%) throughout.

VLA backbones and baselines. We compare raw-action and ViDAL latent targets on the same VLA backbone. Regression policies use OpenVLA-OFT (Kim et al., 2025) on LIBERO and StarVLA-OFT (Ye et al., 2026a) on RoboTwin 2.0. Flow-matching policies use $\pi_{0.5}$ (Intelligence et al., 2025) on both benchmarks. LIBERO also includes action-tokenization and visual-dynamics VLA comparisons (Pertsch et al., 2025; Liu et al., 2025; Bjorck et al., 2025; Cen et al., 2025; Zhang et al., 2025; Bu et al., 2025; Chen et al., 2025b).

Training and evaluation. We train an Action VAE on demonstrations from each benchmark or robot platform, using a frozen Wan VAE (Wan et al., 2025) for future visual targets. Its frozen encoder and decoder then provide the latent action interface for multi-task VLA training and inference. We use D=8 for single-arm settings and D=16 for dual-arm settings, with λ_{dyna}=1 and λ_{KL}=10^{-4}. Additional configurations are in the appendix.

4.2 Main Results

LIBERO. On LIBERO, ViDAL raises the average success rate of OpenVLA-OFT from 96.8% to 97.1% and that of $\pi_{0.5}$ from 96.9% to 98.1% (Table 1). The $\pi_{0.5}$ variant achieves the highest average among the compared methods. Its largest suite-level gain is on Long, where success rises from 92.4% to 95.6%. OpenVLA-OFT shows a smaller overall gain, with its clearest improvement on Spatial from 97.7% to 98.6%. These paired comparisons establish that a dynamics-grounded latent can improve both regression and flow-matching policies. The Long-suite gain suggests particular value when the policy must sustain object interactions across successive action chunks.

RoboTwin 2.0. On RoboTwin 2.0, ViDAL raises $\pi_{0.5}$ success from 54.3% to 65.5% on Clean and from 33.2% to 43.1% on Random (Table 2). StarVLA-OFT also improves by $+5.1$ and $+1.8$ points on the two splits. The full 50-task results in Appendix D show that $\pi_{0.5}$+ViDAL improves 43 tasks on Clean and 42 on Random. On the Random split, success on *Click Alarmclock* rises from 42% to 79%, and on *Open Laptop* from 48% to 77%. The breadth of these gains suggests that grounding actions in their visual consequences captures task-relevant structure that remains useful as scene appearance changes. Improvements with both $\pi_{0.5}$ and StarVLA-OFT further show that this representation serves different action experts.

Table 1: LIBERO task success rates (%). Gray rows use ViDAL with the corresponding VLA backbone. Gains in the Average column are percentage-point differences from that backbone.

Method	Object	Spatial	Goal	Long	Average
Action-tokenization VLAs					
π_0-FAST (Pertsch et al., 2025)	97.2	96.6	96.0	86.8	94.2
FASTer (Liu et al., 2025)	**99.4**	98.0	**98.6**	95.4	97.9
Visual-Dynamics Augmented VLAs					
GR00T-N1.5 (Bjorck et al., 2025)	97.6	94.4	93.0	90.6	93.9
WorldVLA (Cen et al., 2025)	96.2	87.6	83.4	60.0	81.8
DreamVLA (Zhang et al., 2025)	94.0	97.5	89.5	89.5	92.6
UniVLA (Bu et al., 2025)	96.8	96.5	95.6	92.0	95.2
Villa-X (Chen et al., 2025b)	97.0	97.5	91.5	74.5	90.1
VLA backbones (vanilla vs. +ViDAL)					
OpenVLA-OFT (Kim et al., 2025)(1 policy for all 4 suites)	98.0	97.7	96.1	95.3	96.8
OpenVLA-OFT+ViDAL	97.8	98.6	96.2	**95.6**	97.1(+0.3)
$\pi_{0.5}$ (Intelligence et al., 2025)	98.2	98.8	98.0	92.4	96.9
$\pi_{0.5}$+ViDAL	**99.4**	**99.2**	98.2	**95.6**	**98.1**(+1.2)

Table 2: RoboTwin 2.0 task success rates (%) under Clean and Random evaluation. The table shows 15 representative tasks and averages over all 50 tasks. Gray columns use ViDAL. Parenthetical gains are percentage-point differences from the corresponding backbone.

	π_0 (Black et al., 2024)		StarVLA-OFT (Ye et al., 2026a)		StarVLA-OFT +ViDAL		$\pi_{0.5}$ (Intelligence et al., 2025)		$\pi_{0.5}$ +ViDAL	
Task	Clean	Random	Clean	Random	Clean	Random	Clean	Random	Clean	Random
Adjust Bottle	90.0	56.0	96.0	0.0	**100.0**	5.0	84.0	73.0	99.0	**86.0**
Place Empty Cup	37.0	11.0	72.0	4.0	66.0	0.0	80.0	56.0	**89.0**	**83.0**
Click Alarmclock	63.0	11.0	**91.0**	14.0	81.0	6.0	56.0	42.0	79.0	**79.0**
Open Laptop	85.0	46.0	31.0	0.0	79.0	6.0	75.0	48.0	**91.0**	**77.0**
Place Burger Fries	80.0	4.0	96.0	6.0	**97.0**	4.0	68.0	47.0	85.0	**77.0**
Dump Bin Bigbin	83.0	24.0	68.0	1.0	87.0	7.0	86.0	54.0	**92.0**	**69.0**
Move Playingcard Away	53.0	22.0	69.0	1.0	**93.0**	5.0	64.0	35.0	73.0	**68.0**
··· *(50 tasks total)*										
Handover Mic	98.0	13.0	39.0	0.0	90.0	1.0	75.0	26.0	**100.0**	**54.0**
Press Stapler	62.0	29.0	60.0	0.0	78.0	11.0	66.0	50.0	**83.0**	**55.0**
Open Microwave	**80.0**	50.0	54.0	2.0	60.0	15.0	63.0	31.0	75.0	**53.0**
Stack Bowls Three	66.0	24.0	62.0	0.0	32.0	0.0	53.0	44.0	**78.0**	**52.0**
Place Bread Basket	17.0	4.0	52.0	5.0	**61.0**	2.0	41.0	33.0	57.0	**46.0**
Pick Dual Bottles	57.0	12.0	43.0	13.0	63.0	8.0	54.0	**49.0**	**69.0**	42.0
Put Bottles Dustbin	54.0	13.0	1.0	0.0	9.0	1.0	47.0	36.0	**68.0**	**42.0**
Stack Bowls Two	91.0	41.0	90.0	1.0	67.0	0.0	92.0	66.0	**99.0**	**75.0**
Average	46.4	16.3	49.5	2.2	54.6(+5.1)	4.0(+1.8)	54.3	33.2	**65.5**(+11.2)	**43.1**(+9.9)

Table 3: Action VAE training-objective ablation with $\pi_{0.5}$ on RoboTwin 2.0. Clean and Random success rates (%) are averaged over 50 tasks. Every variant includes action reconstruction and KL regularization.

Variant	$\mathcal{L}_{\text{dyna}}$	$\mathcal{L}_{\text{freq}}$	$\mathcal{L}_{\text{KL}}$	Clean	Random
Vanilla Action VAE			✓	57.4	38.3
w/o $\mathcal{L}_{\text{dyna}}$		✓	✓	62.2	40.1
w/o $\mathcal{L}_{\text{freq}}$	✓		✓	63.1	40.4
Full ViDAL	✓	✓	✓	**65.5**	**43.1**

4.3 Ablation Studies

We study the Action VAE objective and latent design using $\pi_{0.5}$ on RoboTwin 2.0. Each variant retrains the Action VAE and is evaluated across all 50 tasks.

Training objective. The vanilla Action VAE reaches $57.4\%/38.3\%$ on Clean/Random, above the raw-action $\pi_{0.5}$ baseline at $54.3\%/33.2\%$ (Tables 3 and 2). The full objective further raises success to $65.5\%/43.1\%$. Removing the visual-dynamics loss lowers success by 3.3/3.0 points, while removing frequency-domain reconstruction lowers it by 2.4/2.7 points. The ablation separates compression from visual grounding: the Action VAE raises success over raw-action prediction, and future visual prediction provides the larger incremental gain within the full objective.

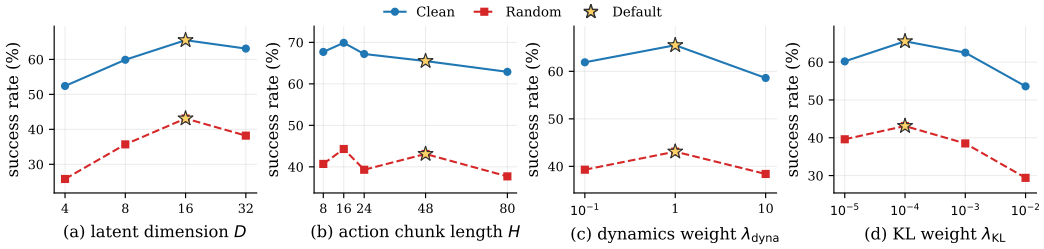


Figure 3: Sensitivity of RoboTwin 2.0 success to Action VAE design with $\pi_{0.5}$. Each panel varies one parameter while holding the others fixed. Stars mark the main-experiment setting.

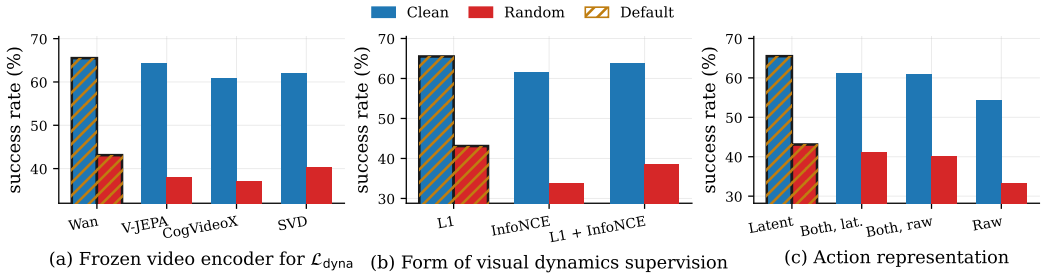


Figure 4: Effects of visual targets and policy action targets on RoboTwin 2.0 with $\pi_{0.5}$. (a) Frozen video encoder providing future visual latents. (b) Visual-dynamics training objective. (c) Policy prediction target. *Both, lat./raw* predicts both targets and decodes the named branch at inference.

Latent design choices. Figure 3 varies latent dimension D, action chunk length H, visual-dynamics weight λ_{dyna}, and KL weight λ_{KL}. The default D=16, λ_{dyna}=1, and λ_{KL}=10^{-4} give the highest success in their respective sweeps. Raising the KL weight to 10^{-2} lowers success to 53.6%/29.4%, showing that the strength of latent regularization affects the learned policy target. The horizon sweep reveals additional headroom: H=16 reaches 69.9%/44.3%, compared with 65.5%/43.1% at the baseline-aligned H=48 used in the main comparisons.

4.4 Further Analysis

We next compare future visual targets and policy prediction targets, then examine how the paired gain changes with training scale.

Visual-dynamics supervision. Dense L1 regression to future Wan VAE latents gives the strongest result among the visual targets and objectives tested (Fig. 4(a,b)). Changing the frozen encoder to V-JEPA yields 64.4%/37.9% on Clean/Random, compared with 65.5%/43.1% for Wan. CogVideoX and SVD targets also produce lower success. With the Wan target fixed, replacing L1 with InfoNCE reduces success to 61.6%/33.8%, and combining the objectives reaches 63.9%/38.6%. The larger Random-split gaps favor feature-level regression of future video latents over a contrastive match, making dense scene-transition prediction the stronger supervision in this setting.

Policy prediction target. The learned latent works best as the sole policy target (Fig. 4(c)). It reaches 65.5%/43.1%, compared with 54.3%/33.2% for a raw-action target. Joint prediction of raw actions and latents reaches 61.2%/41.3% when the latent branch is decoded and 61.0%/40.1% when the raw branch is used. Both are below latent-only prediction. Splitting the action expert across raw and latent targets weakens the benefit of learning directly in the dynamics-grounded latent space.

Together, these ablations support a consistent design. Dense future-visual supervision shapes the Action VAE latent, and the downstream expert predicts that latent alone. The resulting action target serves both regression and flow-matching policies.

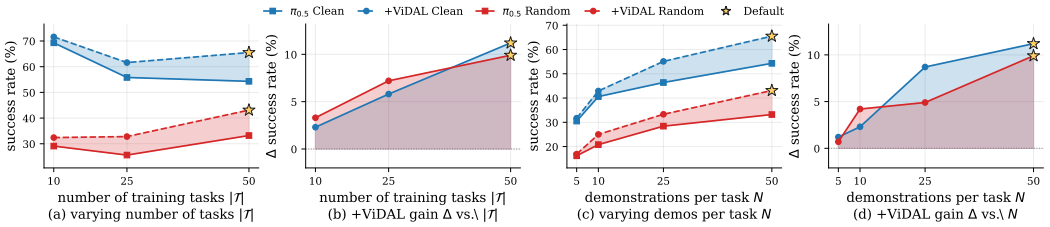


Figure 5: Paired scaling results on RoboTwin 2.0 with $\pi_{0.5}$. (a,b) Success and gain as the training-task count increases, evaluated on the corresponding task subset. (c,d) Success and gain as demonstrations per task increase, evaluated on all 50 tasks. The gain Δ is the percentage-point difference between ViDAL and the raw-action backbone. The task subsets are listed in Appendix C.

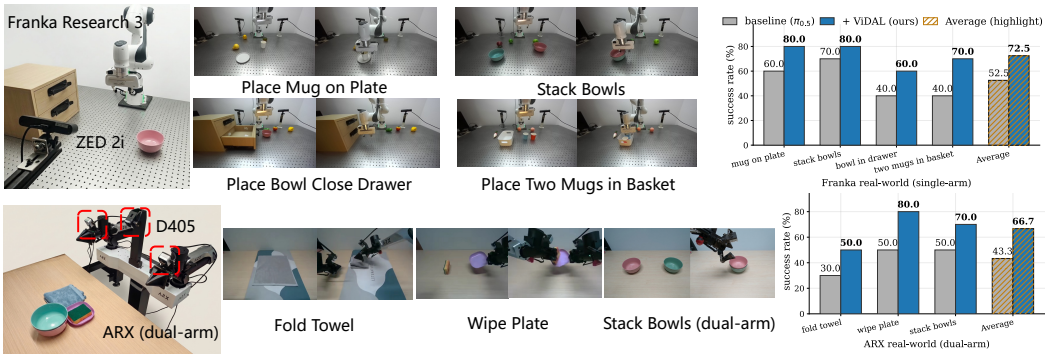


Figure 6: Real-robot evaluation on Franka (top) and ARX (bottom). The figure shows each platform, representative rollouts, and task success rates for $\pi_{0.5}$ with raw-action or ViDAL latent targets. Each task has 10 evaluation rollouts.

Scaling with tasks and demonstrations. ViDAL improves $\pi_{0.5}$ at every tested training scale (Fig. 5). The task-count experiment uses nested 10-, 25-, and 50-task subsets, so the paired gain within each subset is the relevant comparison. On Clean, that gain rises from $+2.3$ to $+5.8$ to $+11.2$ points, and on Random from $+3.3$ to $+7.2$ to $+9.9$. The demonstration-count experiment holds the 50 evaluation tasks fixed. Increasing demonstrations per task from 5 to 50 raises the paired gain from $+1.2$ to $+11.2$ points on Clean and from $+0.7$ to $+9.9$ on Random. More tasks expose the latent to a broader set of scene transitions, while more demonstrations refine the action–outcome relation within each task through repeated observations of the resulting changes in object pose and contact.

4.5 Real-Robot Experiments

Setup. We evaluate on four tabletop tasks with a single-arm Franka Research 3 and three bimanual tasks with an ARX robot (Fig. 6). An Action VAE is trained on each platform's demonstrations, then frozen for $\pi_{0.5}$ training and inference. The two $\pi_{0.5}$ policies on each platform use the same VLM backbone, demonstrations, and training recipe. Only the action target changes from raw commands to ViDAL latents. Each task has 10 evaluation rollouts.

Results. ViDAL improves every evaluated task on both robots (Fig. 6). Average Franka success rises from 52.5% to 72.5%, a $+20.0$-point gain. The four task-level improvements range from $+10$ to $+30$ points, with *Place Two Mugs in Basket* rising from 40% to 70%. Average ARX success rises from 43.3% to 66.7%, a $+23.4$-point gain. All three bimanual tasks improve by at least 20 points, including *Wipe Plate* from 50% to 80%. On Franka, *Place Two Mugs in Basket* requires successive placements as the scene changes after each grasp. Its gain is consistent with an action code that preserves the visual consequences of contact. On ARX, improvement spans deformable *Fold Towel* and rigid *Stack Bowls*, indicating value across distinct object dynamics. The same representation-learning objective applies on both platforms despite their different action spaces and visual transitions.

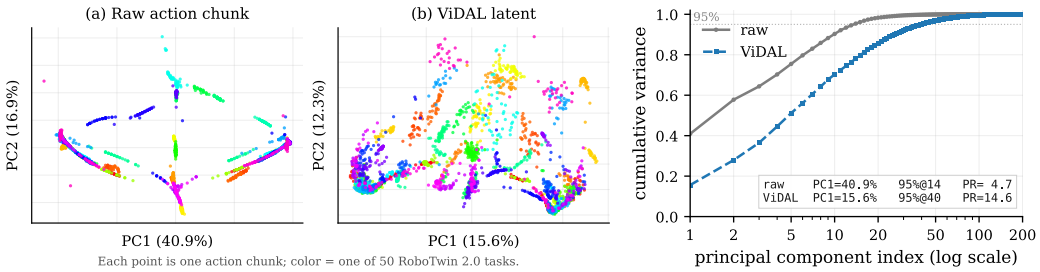


Figure 7: PCA of raw action chunks and ViDAL latents from 50 RoboTwin 2.0 tasks. Left and middle panels show 2D projections colored by task. The right panel shows cumulative explained variance across principal components.

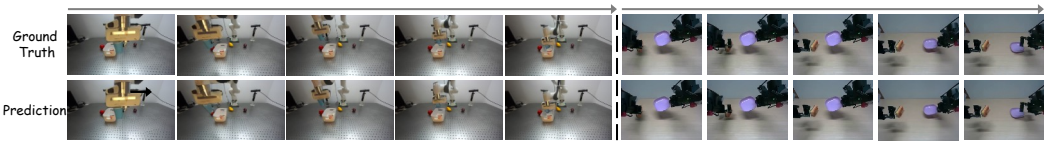


Figure 8: Future video prediction on Franka (left) and ARX (right). Top rows show observed future frames. Bottom rows show frames decoded from the future visual latents predicted using the current video and ViDAL action latent.

4.6 Qualitative Visualization and Analysis

We examine the distribution of learned action latents and visualize the future video frames predicted from them.

Action latent space geometry. The two action spaces distribute variance differently across components (Fig. 7). The first principal component explains 40.9% of raw-action variance and 15.6% of ViDAL latent variance. Reaching 95% cumulative variance takes 14 and 40 components, respectively. The corresponding participation ratios (Gao & Ganguli, 2015) are 4.7 and 14.6. In the 2D projections, raw action chunks occupy a few narrow trajectories, while learned latents spread across more directions. The learned representation thus uses more directions of variation than raw action chunks, consistent with a code shaped by both action reconstruction and future visual change.

Future video prediction. The same action latent predicted by the VLA policy also drives recognizable future video when paired with the current visual state (Fig. 8). Across successive predicted frames, the decoded videos follow object displacement and changes in gripper pose on both Franka and ARX trajectories. These are the observable consequences of the action chunks represented by the latent. Together with the objective ablation, the examples show how future visual prediction shapes a policy target that improves manipulation success.

5 Conclusion

We propose ViDAL, a method that grounds the action latent space in future visual dynamics by jointly training an Action VAE on action reconstruction and video latent prediction. The frozen Action VAE then serves as a plug-in action representation for both regression and flow-matching VLA policies without modifying the VLM backbone. ViDAL consistently improves performance across two simulation benchmarks (LIBERO, RoboTwin 2.0) and two real-robot platforms (single-arm Franka, dual-arm ARX). Further analysis confirms that the advantage widens as tasks and demonstrations scale up. Since ViDAL decouples action-representation learning from VLA policy training, extending it to diverse action spaces and embodiments requires only retraining the lightweight Action VAE. Leveraging large-scale egocentric human videos as an additional source of visual-dynamics supervision is a natural next step toward a universal dynamics-grounded action latent space for generalist policies across robot embodiments and diverse manipulation tasks.

AI USE STATEMENT

We used generative AI tools to aid and polish the writing of this paper, including improving clarity and readability and assisting with LaTeX formatting.

ETHICS STATEMENT

The study evaluates robot manipulation in simulation and on real robots. It raises no specific ethical concerns.

REPRODUCIBILITY STATEMENT

The main paper describes the method, training objectives, experimental setup, and evaluation protocols. The appendix provides further details on the Action VAE architecture and implementation, benchmark-specific training configurations, full RoboTwin 2.0 results, and additional qualitative examples.

REFERENCES

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, et al. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. 2025a.

Xiaoyu Chen, Hangxing Wei, Pushi Zhang, Chuheng Zhang, Kaixin Wang, Yanjiang Guo, Rushuai Yang, Yucen Wang, Xinquan Xiao, Li Zhao, et al. Villa-x: enhancing latent action modeling in vision-language-action models. *arXiv preprint arXiv:2507.23682*, 2025b.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. 2023.

Peiran Gao and Surya Ganguli. On simplicity and complexity in the brave new world of large-scale neuroscience. *Current opinion in neurobiology*, 32:148–155, 2015.

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. 2024.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael 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. *arXiv preprint arXiv:2504.16054*, 2025.

Yueru Jia, Jiaming Liu, Shengbang Liu, Rui Zhou, Wanhe Yu, Yuyang Yan, Xiaowei Chi, Yandong Guo, Boxin Shi, and Shanghang Zhang. Video2act: A dual-system video diffusion policy with robotic spatio-motional modeling. *arXiv preprint arXiv:2512.03044*, 2025.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. 2023.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

Yicheng Liu, Shiduo Zhang, Zibin Dong, Baijun Ye, Tianyuan Yuan, Xiaopeng Yu, Linqi Yin, Chenhao Lu, Junhao Shi, Luca Jiang-Tao Yu, et al. Faster: Toward efficient autoregressive vision language action modeling via neural action tokenization. *arXiv preprint arXiv:2512.04952*, 2025.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. pp. 6892–6903, 2024.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

Shuyao Shang, Bing Zhan, Yunfei Yan, Yuqi Wang, Yingyan Li, Yasong An, Xiaoman Wang, Jierui Liu, Lu Hou, Lue Fan, et al. Dynvla: Learning world dynamics for action reasoning in autonomous driving. *arXiv preprint arXiv:2603.11041*, 2026.

Yue Su, Sijin Chen, Haixin Shi, Mingyu Liu, Zhengshen Zhang, Ningyuan Huang, Weiheng Zhong, Zhengbang Zhu, Yuxiao Liu, and Xihui Liu. World guidance: World modeling in condition space for action generation. *arXiv preprint arXiv:2602.22010*, 2026.

Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, Guangzhong Sun, Xin Jin, and Zhibo Chen. Vla-jepa: Enhancing vision-language-action model with latent world model. *arXiv preprint arXiv:2602.10098*, 2026.

Wenhui Tan, Bei Liu, Junbo Zhang, Ruihua Song, and Jianlong Fu. Rold: Robot latent diffusion for multi-task policy modeling. pp. 340–353, 2024.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. 2024.

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 *Conference on Robot Learning*, pp. 1723–1736. PMLR, 2023.

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.

Yating Wang, Haoyi Zhu, Mingyu Liu, Jiange Yang, Hao-Shu Fang, and Tong He. Vq-vla: Improving vision-language-action models via scaling vector-quantized action tokenizers. pp. 11089–11099, 2025.

Xiaoxu Xu, Hao Li, Jinhui Ye, Yilun Chen, Jia Zeng, Xinyi Chen, Linning Xu, Dahua Lin, Weixin Li, and Jiangmiao Pang. Futurevla: Joint visuomotor prediction for vision-language-action model. *arXiv preprint arXiv:2603.10712*, 2026.

Jinhui Ye, Ning Gao, Senqiao Yang, Jinliang Zheng, Zixuan Wang, Yuxin Chen, Pengguang Chen, Yilun Chen, Shu Liu, and Jiaya Jia. StarVLA-α: Reducing complexity in vision-language-action systems. *arXiv preprint arXiv:2604.11757*, 2026a.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026b.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, Fan Lu, He Wang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*, 2025.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Hongyi Zhou, Weiran Liao, Xi Huang, Yucheng Tang, Fabian Otto, Xiaogang Jia, Xinkai Jiang, Simon Hilber, Ge Li, Qian Wang, et al. Beast: Efficient tokenization of b-splines encoded action sequences for imitation learning. *arXiv preprint arXiv:2506.06072*, 2025.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A METHOD DETAILS

Action VAE architecture. The Action VAE maps a normalized action chunk $\mathbf{a}_{t:t+H-1} \in \mathbb{R}^{H \times d_a}$ to a short sequence of continuous action-latent tokens. Its encoder is a temporal convolutional network with residual blocks and strided temporal downsampling, so the latent sequence preserves the temporal order of the action chunk while reducing its length. The encoder predicts a Gaussian posterior with mean and variance parameters, from which the action latent is sampled during representation learning. For downstream VLA supervision, we use the posterior mean as a deterministic action-latent target. The decoder mirrors the encoder with temporal upsampling and reconstructs the original action chunk from the latent sequence. This design keeps the action representation continuous, temporally structured, and directly decodable into executable robot actions.

Frozen video-latent target. For each action chunk, we use the aligned observation clip $\mathbf{o}_{t:t+H}$, which contains one more frame than the action horizon. A frozen video VAE encodes this clip into spatiotemporal latent tokens. For a clip with $H{+}1 = 4k{+}1$ frames, image size $h{\times}w$, and video-latent channel dimension C_v, the flattened video latent has shape

$$\mathbf{z}_v \in \mathbb{R}^{T_v N_s \times C_v}, \qquad T_v = k+1, \quad N_s = (h/16)(w/16). \tag{8}$$

We split this latent into the first-frame spatial tokens $\mathbf{z}_0 \in \mathbb{R}^{N_s \times C_v}$ and the future target $\mathbf{z}_f \in \mathbb{R}^{(T_v-1)N_s \times C_v}$. The first-frame tokens provide the current visual context, while the future tokens provide the visual-dynamics target induced by the action chunk.

Future-latent predictor. The dynamics predictor learns to infer the future visual latent from the current visual context and the action latent. It represents the current and future visual positions as a shared token sequence, conditions the future queries on the action-latent tokens through cross-attention, and predicts the future visual-latent tokens jointly:

$$\hat{\mathbf{z}}_f = P_\theta(\mathbf{z}_0, \mathbf{z}_a). \tag{9}$$

Conditioning on $\mathbf{z}_0$ disambiguates actions whose visual effects depend on the current scene, while conditioning on $\mathbf{z}_a$ encourages the action latent to encode the scene transition caused by the action chunk. The predictor is used only during representation learning and is not part of the deployed VLA policy.

Training objective. The Action VAE is trained with four terms:

$$\mathcal{L}_{\text{ViDAL}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{freq}}\mathcal{L}_{\text{freq}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}} + \lambda_{\text{dyna}}\mathcal{L}_{\text{dyna}}. \tag{10}$$

The reconstruction term is an L1 loss on normalized actions, $\mathcal{L}_{\text{rec}} = \|\hat{\mathbf{a}} - \mathbf{a}\|_1$. The frequency-domain term applies a discrete cosine transform along the temporal axis and computes an L1 distance between reconstructed and target frequency coefficients. The KL term is the standard Gaussian-posterior regularizer. The visual-dynamics term is an L1 loss on frozen video-latent tokens,

$$\mathcal{L}_{\text{dyna}} = \|P_\theta(\mathbf{z}_0, \mathbf{z}_a) - \text{sg}(\mathbf{z}_f)\|_1, \tag{11}$$

where $\text{sg}(\cdot)$ denotes that the video-latent target is held fixed. The weights and schedules are benchmark-specific and are listed in Section B.

Downstream VLA interface. After representation learning, the video VAE and future-latent predictor are discarded. The frozen Action VAE defines the interface between a VLA policy and the robot: its encoder converts each action chunk into a deterministic latent target, and its decoder converts predicted latents back into executable action chunks. We normalize the action latents before policy training and invert the normalization before decoding at inference. Thus, downstream VLA training changes only the action target space; the VLM backbone and the policy action expert remain otherwise unchanged.

B IMPLEMENTATION DETAILS

Action VAE training. We train a separate Action VAE for each benchmark or robot platform, using the same architecture and objective described in Section A. Table 4 summarizes the main configuration choices. Single-arm settings use the LIBERO configuration, and the Franka real-robot setting follows the same configuration. Dual-arm settings use the RoboTwin 2.0 configuration, and

Table 4: Key Action VAE training configurations. We report only the settings that define the action-latent interface and visual-dynamics target.

Setting	LIBERO / Franka	RoboTwin 2.0 / ARX
Action dimension d_a	7	14
Action chunk length H	32	48
Action latent dimension D	8	16
Observation clip length	33 frames	49 frames
Image resolution	128×128	96×128
Video latent tokens $T_v \times N_s$	9×64	13×48
Training steps	50k	50k

the ARX real-robot setting follows the same configuration. Across settings, we use a frozen Wan video VAE as the visual-dynamics target, $\lambda_{\text{dyna}}=1$, $\lambda_{\text{freq}}=1$, and $\lambda_{\text{KL}}=10^{-4}$.

Downstream VLA integration. For all downstream policies, ViDAL only changes the action representation: the raw action chunk target is replaced by the normalized Action VAE latent, and the predicted latent is decoded by the frozen Action VAE decoder before execution. The original VLA training and inference recipes are otherwise kept unchanged for OpenVLA-OFT and StarVLA-OFT. For $\pi_{0.5}$, we keep the original flow-matching objective and inference procedure, but set the action expert learning rate to $10\times$ the VLM learning rate. This adjustment helps the pretrained action expert adapt from the raw-action distribution used in $\pi_{0.5}$ pretraining to the ViDAL latent distribution. At inference, neither the video VAE nor the future-latent predictor is used; the only additional deployed module is the Action VAE decoder, whose latency overhead is negligible compared with the VLA forward pass.

Training data and evaluation. On LIBERO, we use the official training demonstrations and official evaluation protocol. On RoboTwin 2.0, we train on the Clean demonstrations from 50 tasks, with 50 episodes per task and 2,500 episodes in total; evaluation follows the official Clean and Random settings, with 100 rollouts per task in each setting. For real-robot experiments, the single-arm Franka setting contains four tasks with 50 demonstrations per task, and the dual-arm ARX setting contains three tasks with 100 demonstrations per task. Each real-robot task is evaluated with 10 rollouts, and a rollout is counted as successful if the robot completes the language instruction for that task.

C TASK SUBSETS FOR SCALING

The 10- and 25-task settings in Fig. 5(a,b) use nested RoboTwin 2.0 subsets, with training and evaluation on the same selected tasks. Table 5 lists their task identifiers. The 50-task setting uses the full task set in Table 6.

Table 5: RoboTwin 2.0 task subsets for task-count scaling. The 25-task set contains the 10-task set.

10-task subset	25-task subset	
adjust_bottle	adjust_bottle	place_a2b_left
beat_block_hammer	beat_block_hammer	place_a2b_right
blocks_ranking_rgb	blocks_ranking_rgb	place_bread_skillet
click_alarmclock	click_alarmclock	place_burger_fries
handover_block	click_bell	place_can_basket
lift_pot	handover_block	place_container_plate
open_laptop	hanging_mug	place_dual_shoes
pick_diverse_bottles	lift_pot	place_object_basket
shake_bottle	move_can_pot	rotate_qrcode
stack_blocks_two	move_pillbottle_pad	shake_bottle
	open_laptop	stack_blocks_two
	pick_diverse_bottles	stack_bowls_two
	pick_dual_bottles	

D FULL ROBOTWIN 2.0 RESULTS

Table 6 reports the full per-task RoboTwin 2.0 results corresponding to Table 2. All methods are evaluated under both Clean and Random settings, with 100 rollouts per task in each setting. The representative rows shown in the main paper are a subset of this full table, and the averages match Table 2 up to rounding.

E FUTURE PREDICTION EXAMPLES

Figures 9 and 10 show additional future visual prediction examples from the learned action latent across simulation and real-robot data. For each example, the top row shows ground-truth future frames from the trajectory, and the bottom row shows frames decoded from the predicted future video latent conditioned on the ViDAL action latent. These visualizations are used only as qualitative analysis of the latent space; the video VAE, future-latent predictor, and video decoder are not part of the deployed policy.

Table 6: Full RoboTwin 2.0 success rates (%) over all 50 tasks. Gray columns indicate the corresponding +ViDAL variants.

	π_0		StarVLA-OFT		StarVLA-OFT +ViDAL		$\pi_{0.5}$		$\pi_{0.5}$+ViDAL	
Task	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.
Adjust Bottle	90.0	56.0	96.0	0.0	100.0	5.0	84.0	73.0	99.0	86.0
Beat Block Hammer	43.0	21.0	58.0	1.0	68.0	2.0	68.0	19.0	86.0	28.0
Blocks Ranking RGB	19.0	5.0	45.0	0.0	9.0	0.0	48.0	22.0	71.0	41.0
Blocks Ranking Size	7.0	1.0	27.0	0.0	3.0	0.0	26.0	7.0	37.0	11.0
Click Alarmclock	63.0	11.0	91.0	14.0	81.0	6.0	56.0	42.0	79.0	79.0
Click Bell	44.0	3.0	94.0	4.0	98.0	16.0	64.0	77.0	53.0	51.0
Dump Bin Bigbin	83.0	24.0	68.0	1.0	87.0	7.0	86.0	54.0	92.0	69.0
Grab Roller	96.0	80.0	93.0	14.0	92.0	20.0	96.0	87.0	100.0	89.0
Handover Block	45.0	8.0	0.0	0.0	27.0	0.0	32.0	1.0	49.0	13.0
Handover Mic	98.0	13.0	39.0	0.0	90.0	1.0	75.0	26.0	100.0	54.0
Hanging Mug	11.0	3.0	15.0	0.0	4.0	0.0	9.0	8.0	16.0	11.0
Lift Pot	84.0	36.0	0.0	0.0	56.0	0.0	58.0	2.0	89.0	4.0
Move Can Pot	58.0	21.0	50.0	0.0	71.0	2.0	30.0	11.0	62.0	28.0
Move Pillbottle Pad	21.0	1.0	54.0	0.0	72.0	0.0	35.0	17.0	47.0	23.0
Move Playingcard Away	53.0	22.0	69.0	1.0	93.0	5.0	64.0	35.0	73.0	68.0
Move Stapler Pad	0.0	2.0	12.0	0.0	5.0	0.0	23.0	12.0	12.0	13.0
Open Laptop	85.0	46.0	31.0	0.0	79.0	6.0	75.0	48.0	91.0	77.0
Open Microwave	80.0	50.0	54.0	2.0	60.0	15.0	63.0	31.0	75.0	53.0
Pick Diverse Bottles	27.0	6.0	30.0	7.0	42.0	8.0	51.0	40.0	63.0	30.0
Pick Dual Bottles	57.0	12.0	43.0	13.0	63.0	8.0	54.0	49.0	69.0	42.0
Place A2B Left	31.0	1.0	20.0	0.0	62.0	2.0	66.0	26.0	62.0	39.0
Place A2B Right	27.0	6.0	22.0	2.0	44.0	3.0	52.0	17.0	59.0	35.0
Place Bread Basket	17.0	4.0	52.0	5.0	61.0	2.0	41.0	33.0	57.0	46.0
Place Bread Skillet	23.0	1.0	56.0	2.0	52.0	2.0	36.0	21.0	52.0	29.0
Place Burger Fries	80.0	4.0	96.0	6.0	97.0	4.0	68.0	47.0	85.0	77.0
Place Can Basket	41.0	5.0	63.0	0.0	68.0	1.0	52.0	3.0	62.0	12.0
Place Cans Plasticbox	34.0	2.0	81.0	2.0	87.0	3.0	28.0	32.0	19.0	33.0
Place Container Plate	88.0	45.0	99.0	3.0	94.0	2.0	95.0	50.0	94.0	76.0
Place Dual Shoes	15.0	0.0	28.0	1.0	2.0	0.0	33.0	10.0	23.0	7.0
Place Empty Cup	37.0	11.0	72.0	4.0	66.0	0.0	80.0	56.0	89.0	83.0
Place Fan	20.0	10.0	28.0	0.0	20.0	0.0	57.0	24.0	44.0	25.0
Place Mouse Pad	7.0	1.0	9.0	0.0	14.0	0.0	24.0	23.0	31.0	25.0
Place Object Basket	16.0	2.0	40.0	1.0	46.0	0.0	65.0	15.0	72.0	19.0
Place Object Scale	10.0	0.0	19.0	0.0	32.0	1.0	54.0	28.0	60.0	38.0
Place Object Stand	36.0	11.0	48.0	2.0	83.0	0.0	77.0	31.0	80.0	55.0
Place Phone Stand	35.0	7.0	24.0	0.0	63.0	1.0	34.0	35.0	58.0	39.0
Place Shoe	28.0	6.0	63.0	0.0	14.0	1.0	58.0	45.0	64.0	42.0
Press Stapler	62.0	29.0	60.0	0.0	78.0	11.0	66.0	50.0	83.0	55.0
Put Bottles Dustbin	54.0	13.0	1.0	0.0	9.0	1.0	47.0	36.0	68.0	42.0
Put Object Cabinet	68.0	18.0	35.0	0.0	34.0	0.0	25.0	31.0	41.0	28.0
Rotate QRCode	68.0	15.0	50.0	0.0	45.0	0.0	64.0	23.0	71.0	23.0
Scan Object	18.0	1.0	13.0	1.0	30.0	1.0	16.0	14.0	32.0	18.0
Shake Bottle	99.0	51.0	98.0	11.0	100.0	23.0	95.0	87.0	100.0	98.0
Shake Bottle Horizontally	97.0	60.0	98.0	10.0	100.0	36.0	96.0	94.0	99.0	99.0
Stack Blocks Three	17.0	0.0	41.0	0.0	13.0	0.0	37.0	0.0	59.0	26.0
Stack Blocks Two	42.0	1.0	83.0	0.0	67.0	1.0	59.0	15.0	83.0	43.0
Stack Bowls Three	66.0	24.0	62.0	0.0	32.0	0.0	53.0	44.0	78.0	52.0
Stack Bowls Two	91.0	41.0	90.0	1.0	67.0	0.0	92.0	66.0	99.0	75.0
Stamp Seal	3.0	4.0	27.0	0.0	25.0	0.0	25.0	11.0	51.0	31.0
Turn Switch	27.0	23.0	26.0	0.0	25.0	6.0	24.0	31.0	39.0	15.0
Average	46.4	16.3	49.5	2.2	54.6	4.0	54.3	33.2	**65.5**	**43.1**

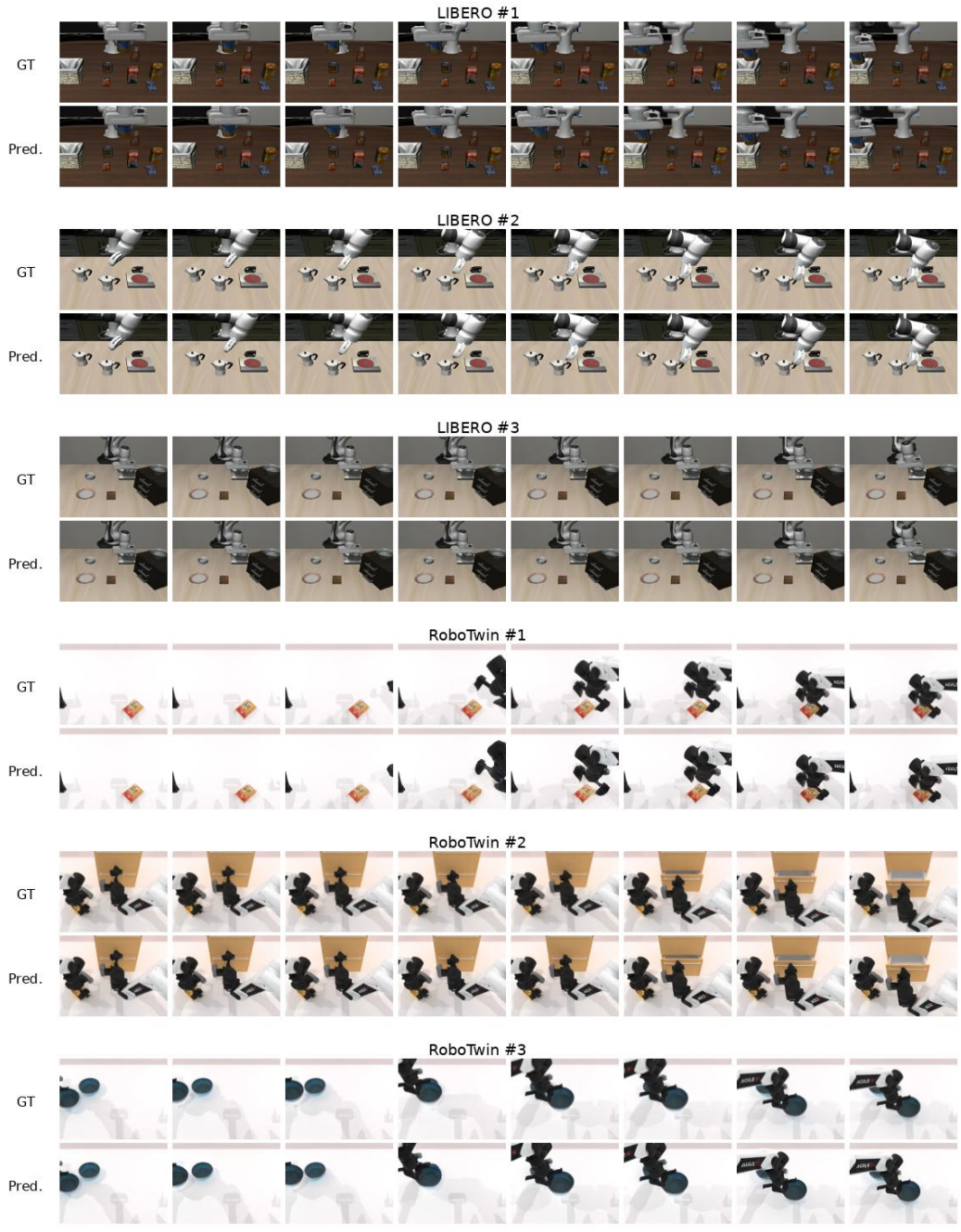


Figure 9: Future visual prediction examples on simulation benchmarks (LIBERO and RoboTwin 2.0). Each example contains a ground-truth row and a prediction row decoded from the future visual latent predicted using the ViDAL action latent.

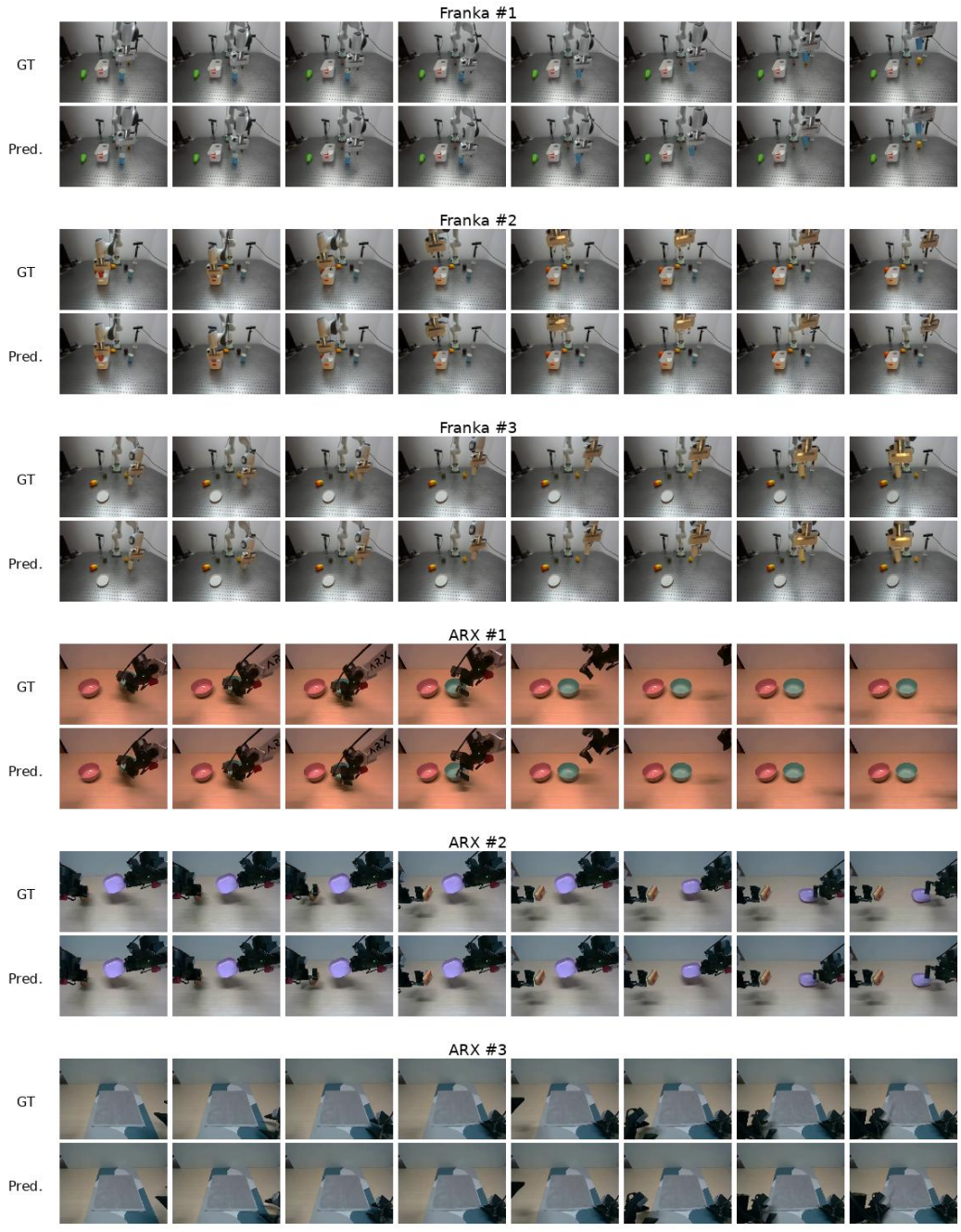


Figure 10: Future visual prediction examples on real-robot datasets (Franka and ARX). Each example contains a ground-truth row and a prediction row decoded from the future visual latent predicted using the ViDAL action latent.