ONE TOKEN PER FRAME: RECONSIDERING VISUAL BANDWIDTH IN WORLD MODELS FOR VLA POLICY

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) models can use visual prediction to anticipate future states, but dense visual features make the generative sequence grow with the number of camera views, prediction horizon, and encoder resolution. Whether such dense representations are necessary for effective control remains unclear. We introduce **OneWM-VLA**, which represents each retained camera view with one predictive token per future step. Adaptive Attention Pooling compresses visual features into compact latents, which are jointly generated with robot actions under a conditional flow-matching objective. Future observations provide the latent targets during training and are not required at inference. This design incorporates visual prediction into a pretrained VLA policy while keeping the generative sequence compact. On MetaWorld MT50, OneWM-VLA improves the average success rate of the π_0 backbone from 47.91% to 61.53%, reaching 72.01% after 60k training steps. It also achieves 98.1% success on LIBERO and raises *Fold Cloth* success on a real Piper arm from 20.0% to 60.0% relative to π_0. Comparisons on two additional VLA backbones consistently favor one token over three across the evaluated checkpoints. A matched ablation at a longer action horizon further shows that removing the latent loss reduces success from 58.09% to 21.64%, supporting the benefit of future supervision for policy learning.

1 INTRODUCTION

Vision-language-action (VLA) models map visual observations, language instructions, and robot states to actions. Recent systems Brohan et al. (2022); Zitkovich et al. (2023); Kim et al. (2024; 2025); Chi et al. (2025); Black et al. (2026); Intelligence et al. (2025) show that large multimodal backbones can support many language-conditioned manipulation tasks. Long-horizon control, however, requires more than reacting to the current observation. A policy must retain task-relevant state, anticipate changes in contacts and object configurations, and keep local errors from accumulating over time. This has motivated VLA policies that augment action generation with predicted frames, latent states, or other future targets Cen et al. (2025b); Zhang et al. (2025b); Hu et al. (2024).

Prediction creates a basic design tradeoff. Dense future features preserve spatial detail, but every retained view contributes another block of visual tokens at every future step. When one transformer co-generates predictions and actions, every additional predictive token participates in each sampling step. The sequence length therefore grows with both the prediction horizon and the encoder resolution. This cost is especially relevant when the predictive stream is added to a pretrained VLA under a limited adaptation budget. Recurrent state-space models Hafner et al. (2019; 2025) and joint-embedding predictors LeCun (2022); Bardes et al. (2024) suggest that useful future information need not take the form of dense reconstruction, but they do not determine how much visual information a generative VLA should retain for control.

We refer to this allocation as *predictive visual bandwidth*: the number of visual tokens assigned to each retained view and future step. This bandwidth controls both the capacity of the predictive representation and the length of the generated suffix. An overly narrow representation may discard task-relevant changes, whereas a wide representation may increase memory and sampling cost without improving the action policy. This makes the operating point an empirical design choice for the joint visual–action generator, rather than a direct consequence of the encoder patch count. Our central

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.

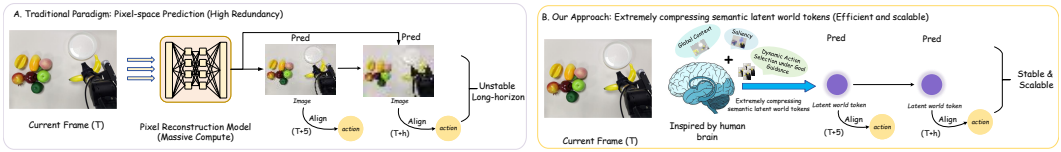

Figure 1: OneWM-VLA represents each retained camera view at each future step with one predictive token. Total suffix length grows with the horizon, while the per-step predictive bandwidth is independent of the encoder patch count.

question is whether a single predictive token per view can retain useful future structure while keeping the generated trajectory compact.

We introduce **OneWM-VLA** (Fig. 1) to study this question. Adaptive Attention Pooling compresses the encoder features of each retained camera view into one token per frame. During training, the same module extracts targets from future frames, producing a compact latent trajectory aligned with the demonstrated actions. A conditional flow then generates the latent and action trajectories together. The interleaved sequence places each future latent block before its action block, allowing the action stream to use the current latent estimate while both trajectories are generated. The latent trajectory remains internal to the policy, while only the generated actions are executed. We initialize the policy from π_0 Black et al. (2026) and adapt the backbone with LoRA Hu et al. (2022) together with the pooling and projection modules.

We evaluate OneWM-VLA at three levels. At the policy level, it improves the average success rate of π_0 from 47.91% to 61.53% under the primary MetaWorld MT50 protocol, reaches 98.1% on LIBERO, and, relative to π_0, raises success on the real-world *Fold Cloth* task from 20.0% to 60.0%. At the design level, we study predictive bandwidth, longer training, pooling, target construction, latent supervision, and action horizon, with additional bandwidth comparisons on $\pi_{0.5}$ and SmolVLA. At the representation level, we test whether the latent representation collapses and whether it retains task structure. Together, these experiments evaluate the performance and cost of the compact interface across training durations and backbones.

Our contributions are as follows:

- We formulate predictive visual bandwidth as a design choice in generative VLA policies and study its relation to sequence length and control performance.
- We introduce OneWM-VLA, which compresses each retained view into one predictive token and co-generates the latent and action trajectories in the same flow.
- We evaluate the design across three manipulation settings, with longer training, additional VLA backbones, alternative target constructors, and representation diagnostics.

2 Related Work

2.1 Vision-Language-Action Models

Progress in multimodal language models Beyer et al. (2024); Li et al. (2023); Wang et al. (2024); Bai et al. (2025b;a) and large robot datasets has led to VLA policies that map images, instructions, and robot state to actions Brohan et al. (2022); Zitkovich et al. (2023); Kim et al. (2024); Intelligence et al. (2025); Zhao et al. (2023); Chen et al. (2025); Black et al. (2026); Tang et al. (2025b); Qu et al. (2025); Bu et al. (2025); Zheng et al. (2025); Kim et al. (2025); Goyal et al. (2025); Team et al. (2024); Shukor et al. (2025). Diffusion and flow-matching policies Chi et al. (2025); Black et al. (2026); Intelligence et al. (2025); Liu et al. (2024) model multimodal action distributions; other methods add reasoning traces, auxiliary heads, or intermediate multimodal outputs Zhao et al. (2025); Ye et al. (2025); Zeng et al. (2025); Huang et al. (2025); Team et al. (2025); Li et al. (2024b). OneWM-VLA retains the existing flow-based action generator and studies how much bandwidth to allocate to its predictive visual stream.

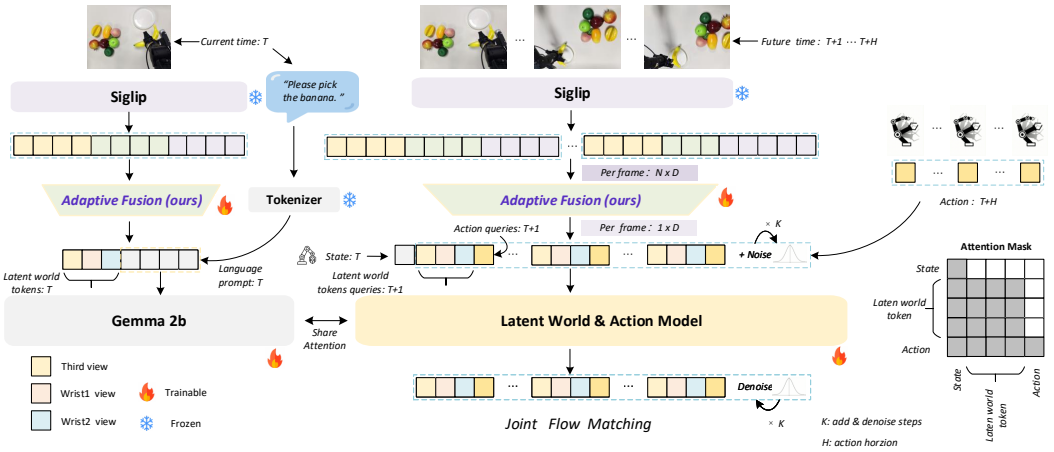

Figure 2: **OneWM-VLA.** Adaptive Attention Pooling compresses the base and left-wrist views. The expert samples interleaved latent and action blocks with a shared flow time; only the actions are executed.

2.2 Predictive Representations and Latent Planning

Compact visual representations and learned rewards have been widely studied for robotic control Nair et al. (2022); Ma et al. (2022; 2023); Li et al. (2024a). Pixel- and video-space methods predict future observations as goals or auxiliary targets Cen et al. (2025b); Bjorck et al. (2025); Cheang et al. (2024); Zhen et al. (2024); Xiao et al. (2025); Du et al. (2023); Jang et al. (2025), while subgoal generation and latent planning provide future guidance for downstream policies Xie et al. (2025); Liu et al. (2025); other systems generate visual futures and actions in a shared model Guo et al. (2024); Zhang et al. (2025a); Hu et al. (2024); Cen et al. (2025a); Jiang et al. (2025); Zhang et al. (2025b). Latent state-space models Hafner et al. (2019; 2025), joint-embedding prediction LeCun (2022); Bardes et al. (2024); Assran et al. (2023); Maes et al. (2026), and sequence models Chen et al. (2021); Tang et al. (2025a) show that useful prediction need not reconstruct every pixel.

Across these lines of work, predictive targets range from dense observations to compact learned states, and joint visual–action generation is already established. OneWM-VLA studies a specific interface within a pretrained flow-based VLA: one predictive token per retained view and future step. Our contribution is the design and empirical evaluation of this compact configuration, including its measured cost and its dependence on target construction, training duration, and backbone.

3 Method

Flow matching learns a velocity field along a path between data and noise, and its conditional form provides a closed-form regression target without ODE integration during training Lipman et al. (2022). OneWM-VLA retains the straight path of the π_0 action generator and extends it to visual latents. The method has two components (Fig. 2). Adaptive Attention Pooling maps each retained view to one latent target token (Sec. 3.1), and a shared flow generates the future latent and action trajectories together (Sec. 3.2). We use the L_1 velocity-regression loss selected in Appendix C.

3.1 Per-Frame Compression via Adaptive Attention Pooling

Adaptive Attention Pooling reduces each retained camera view to one token per frame (Fig. 3). For view i, the pretrained PaliGemma encoder Beyer et al. (2024) produces $N{=}256$ patch features, denoted by $\{\mathbf{x}_i^{(n)}\}_{n=1}^{N}$. Each feature lies in $\mathbb{R}^D$, with $D{=}2048$. The pooling module scores these features in three complementary ways:

$$\phi_{i,\textsc{Max}}(\mathbf{x}) = \max_d x^{(d)}, \quad \phi_{i,\textsc{Sum}}(\mathbf{x}) = \sum_{d=1}^{D} x^{(d)}, \quad \phi_{i,\textsc{Learn}}(\mathbf{x}) = Q_{\theta_i}(\mathbf{x}). \tag{1}$$

where Q_{θ_i} is a small view-specific MLP. Within each branch, a softmax over these scores at temperature τ_p gives the patch weights. The MAX branch applies a coordinate-wise maximum to the weighted features, while the SUM and LEARN branches use weighted sums. This produces three pooled vectors $\mathbf{p}_{i,m}$.

Learned logits $\alpha_{i,m}$ define the branch weights that combine these vectors into the final token:

$$\mathbf{Z}_i = \sum_{m \in \mathcal{M}} \beta_{i,m} \mathbf{p}_{i,m}, \qquad \beta_{i,m} = \frac{\exp(\alpha_{i,m}/\tau_p)}{\sum_{m' \in \mathcal{M}} \exp(\alpha_{i,m'}/\tau_p)}, \tag{2}$$

where $\mathcal{M} = \{\text{MAX}, \text{SUM}, \text{LEARN}\}$. We apply the module independently to the base and left-wrist views, denoted by $\mathcal{V} = \{b, w\}$. The patch weights adapt to each frame, while the branch weights are learned across the training set. The result is one compact token per retained view and frame, independent of the number of encoder patches. Across future steps, these tokens form the clean latent sequences used below. In the direct-K bandwidth comparisons, K counts the fused output tokens actually passed to the model per view and frame; branch fusion remains enabled at every token budget.

3.2 JOINT LATENT–ACTION FLOW MATCHING

A classical transition model predicts $p(s_{t+1} \mid s_t, a_t)$. OneWM-VLA instead treats both future actions and visual latents as generated variables and models

$$p_\theta(a, \{z_i\}_{i \in \mathcal{V}} \mid o_t, \ell, s_t)\,. \tag{3}$$

This defines a conditional flow over aligned action and latent sequences. The two streams share one flow time and are updated synchronously. At each future step, the action block can use the current latent iterates without waiting for a completed latent rollout. This differs from a two-stage latent planner followed by inverse dynamics: no observed future state or latent is available at inference, and action updates do not wait for the latent trajectory to finish.

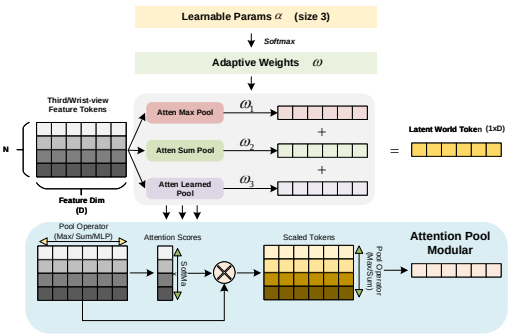

Figure 3: **Adaptive Attention Pooling.** Three scoring branches aggregate patch features from each view, and learned branch weights fuse the pooled vectors into one predictive token.

During training, the same visual encoder and pooling module are applied to the corresponding future frames to construct clean latent targets; future frames are never inference inputs. Let a denote the future action sequence and z_i the latent sequence for view i. These latent variables enter the expert as continuous features. Following the straight path used by π_0 Black et al. (2026), the action and latent variables share one sampled flow time:

$$\begin{aligned} x_t^a &= t\epsilon_a + (1-t)a, \quad x_t^{z_i} = t\epsilon_{z_i} + (1-t)z_i, \\ t &\sim \text{Beta}(1.5, 1). \end{aligned} \tag{4}$$

where the noise terms are sampled independently from a standard Gaussian. The target velocities are $u_t^a = \epsilon_a - a$ and $u_t^{z_i} = \epsilon_{z_i} - z_i$.

The pooled current-view tokens and language tokens form the conditioning prefix, followed by the current robot-state block; neither attends to the generated suffix. The suffix interleaves latent and action variables by future step, with the latent queries placed before the action query at each step. Each latent block attends to the conditioning blocks, latent blocks through the same future step, and action blocks at earlier steps. Each action query attends to the conditioning blocks, all earlier suffix blocks, its own noisy iterate, and the latent iterates at the same step. Training and inference use the same block-visibility pattern.

A shared transformer predicts the action and latent velocity fields with the joint objective

$$\mathcal{L} = \lambda_a \, \mathbb{E}[\|v_\theta^a - u_t^a\|_1] + \sum_{i \in \mathcal{V}} \lambda_i \, \mathbb{E}[\|v_\theta^{z_i} - u_t^{z_i}\|_1]\,. \tag{5}$$



Figure 4: Evaluation suites used in this work: the LIBERO and MetaWorld MT50 simulation benchmarks, and a real Piper robot arm.

The latent and action streams share the generator, flow time, and attention layers, so their predictions evolve together during ODE sampling. View-specific linear input and output projections connect each latent stream to the expert, while a separate head predicts action velocity. Thus, the latent and action variables are updated in one conditional flow, while the visibility mask allows each action block to use the corresponding latent iterates.

Future-target supervision. The latent velocity loss is the only objective that directly supervises the future-latent stream. Its clean endpoints are extracted from the corresponding observed future frames, tying each latent target to a specific future observation and time step. We examine target variation and task separation through covariance-spectrum and feature-space diagnostics in Sec. 4.3.3.

At inference, let $c = (o_t, \ell, s_t)$ denote the observed context and let $\mathbf{x}_t$ collect all action and view-specific latent variables. Starting from $\mathbf{x}_1 \sim \mathcal{N}(0, I)$, we solve

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = v_\theta(t, \mathbf{x}_t \mid c) \tag{6}$$

from $t{=}1$ to $t{=}0$. Only the action component of $\mathbf{x}_0$ is executed; the generated latent trajectory remains internal to the policy.

4 Experiments

We evaluate our method on MetaWorld MT50, LIBERO, and a real Piper arm. MetaWorld is the primary benchmark, with a five-step action horizon used for the primary π_0 experiments unless stated otherwise. We then organize the remaining evidence as ablations and diagnostics of predictive bandwidth, target construction, latent supervision, representation quality, and temporal robustness.

4.1 Experimental Setup

Benchmarks. **MetaWorld MT50** McLean et al. (2025) contains 50 manipulation tasks, grouped into Easy, Medium, Hard, and Very Hard tiers (Appendix A). We evaluate each task for 10 episodes, average task success rates within each tier, and report the unweighted mean of the four tier rates. This convention applies to all MetaWorld MT50 average success rates in the main text and appendix. A separate study evaluates longer action horizons of up to 30 steps. **LIBERO** Liu et al. (2023) contains the Spatial, Object, Goal, and Long suites, with Long emphasizing extended-horizon manipulation. Overall success is the unweighted mean of the four suite success rates. **Real-world Piper** uses a 6-DoF arm with a parallel gripper. We evaluate *Pick Banana*, *Fold Cloth*, and *Pull Drawer* with 20 trials per task in each of two conditions: clean observations and combined perturbations involving lighting changes, object shifts, and unseen distractors. For each condition, overall success is the mean of the three task success rates. Training and inference use action chunks of 50 steps. Figure 4 summarizes the three evaluation settings.

Implementation. We build OneWM-VLA on π_0 Black et al. (2026) and use LoRA Hu et al. (2022) for adaptation. Dense non-LoRA Gemma weights remain frozen, while SigLIP, pooling, and projection modules are trained along with the LoRA adapters. Sampling uses 10 ODE steps; the action loss has weight 1.0, and each latent loss has weight 0.1. The primary MetaWorld models are trained for 30k steps on 8 NVIDIA A800 GPUs. For the main LIBERO evaluation, all suites share

Table 1: MetaWorld MT50 success rates (%). Δ_{π_0} is the percentage-point gain over π_0 at the same horizon. The five-step setting is our primary protocol.

Method	Horizon	Easy	Medium	Hard	Very Hard	Avg%	Δ_{π_0}
Diff. Policy	5	8.21	0.00	0.00	0.00	2.05	–
ACT	5	31.43	11.82	0.00	4.00	11.81	–
π_0	5	71.79	**48.18**	41.67	30.00	47.91	–
OneWM-VLA	5	**79.29**	41.82	**65.00**	**60.00**	**61.53**	**+13.62**
Diff. Policy	10	9.64	0.00	0.00	0.00	2.41	–
ACT	10	47.50	7.27	0.00	4.00	14.69	–
π_0	10	77.50	30.91	30.00	**48.00**	46.60	–
OneWM-VLA	10	**78.57**	**31.82**	**58.33**	40.00	**52.18**	**+5.58**
Diff. Policy	25	8.93	0.00	0.00	0.00	2.23	–
ACT	25	31.43	0.00	0.00	0.00	7.86	–
π_0	25	73.93	27.27	**36.67**	22.00	39.97	–
OneWM-VLA	25	**74.64**	**30.00**	25.00	**54.00**	**45.91**	**+5.94**
Diff. Policy	30	6.43	0.00	0.00	0.00	1.61	–
ACT	30	31.43	0.00	0.00	0.00	7.86	–
π_0	30	**80.00**	**35.45**	35.00	4.00	38.61	–
OneWM-VLA	30	72.86	33.64	**45.00**	**60.00**	**52.87**	**+14.26**

one checkpoint trained with an action horizon of 20; inference settings are selected on a held-out validation set (Appendix D).

Baselines. Our primary backbone baseline is π_0 Black et al. (2026). The LIBERO and real-robot comparisons also include $\pi_{0.5}$ Intelligence et al. (2025). We also report available results for Diffusion Policy Chi et al. (2025), ACT Zhao et al. (2023), Octo Team et al. (2024), OpenVLA Kim et al. (2024), SpatialVLA Qu et al. (2025), CoT-VLA Zhao et al. (2025), WorldVLA Cen et al. (2025b), GR00T-N1 Bjorck et al. (2025), and DreamVLA Zhang et al. (2025b). For methods drawn from prior work, we retain the values in their published evaluation records. Paired ablations use a common training and evaluation protocol within each study.

4.2 Overall Performance

Table 2: Main results on LIBERO. Success rates (%) on the four suites. Inference-time configuration is detailed in Appendix D.

Method	Spatial	Object	Goal	Long	Avg%
Diffusion Policy	78.3	92.5	68.3	50.5	72.4
Octo	78.9	85.7	84.6	51.1	75.1
OpenVLA	84.7	88.4	79.2	53.7	76.5
SpatialVLA	88.2	89.9	78.6	55.5	78.1
CoT-VLA	87.5	91.6	87.6	69.0	83.9
WorldVLA	87.6	96.2	83.4	60.0	81.8
SmolVLA	93.0	94.0	91.0	77.0	88.8
GR00T-N1	94.4	97.6	93.0	90.6	93.9
DreamVLA	97.5	94.0	89.5	89.5	92.6
π_0	96.8	98.8	95.8	85.2	94.2
$\pi_{0.5}$	**98.8**	98.2	98.0	92.4	96.9
OneWM-VLA	98.2	**99.6**	**99.0**	**95.6**	**98.1**

Table 3: Real-world Piper manipulation results. Success rates (%) on three tasks under clean observations and combined perturbations.

Method	P.Banana	F.Cloth	P.Drawer	Avg%
Clean conditions				
π_0	**100.0**	20.0	30.0	50.0
$\pi_{0.5}$	**100.0**	25.0	50.0	58.3
OneWM-VLA	**100.0**	**60.0**	**55.0**	**71.7**
With observation noise				
π_0	65.0	0.0	10.0	25.0
$\pi_{0.5}$	60.0	10.0	25.0	31.7
OneWM-VLA	**75.0**	**40.0**	**35.0**	**50.0**

MetaWorld MT50. At five steps, our method attains 61.53% average success, a 13.62-point gain over π_0 (Tab. 1). The gain is concentrated on the Hard and Very Hard tiers; the Medium tier decreases from 48.18% to 41.82%, so the aggregate gain does not extend to every task group. The remaining rows use matched non-primary horizons and are analyzed separately in Sec. 4.3.4.

Table 4: **Predictive bandwidth ablation** on MetaWorld MT50 ($H = 5$, 30k training steps). K is the number of tokens per view and frame. The $K = 1$ result uses the primary $H = 5$ setting in Tab. 1; OOM denotes out of memory.

K	Total tokens	Trainable (M)	Memory	Latency (ms)	Avg. Success (%)
256 (Full)	$> 2{,}500$	488.83	OOM	–	–
6	126	482.78	Stable	52.95	59.43
3	90	482.76	Stable	52.47	60.63
1	**66**	**482.73**	**Stable**	**44.69**	**61.53**

LIBERO. OneWM-VLA achieves 98.1% average success, above π_0 and $\pi_{0.5}$, with its clearest advantage on the Long suite (Tab. 2). The same checkpoint is used across all four suites; Appendix D lists the fixed inference settings. The Object and Goal suites are close to saturation for these three models, whereas the Long suite leaves a clearer margin. The larger margin on Long is consistent with the intended use of the design in extended interactions; the comparison evaluates the full policy and its validation-selected inference settings.

Real-world Piper. Relative to π_0, OneWM-VLA raises the clean-condition average from 50.0% to 71.7%, with the largest gain on *Fold Cloth*, where success rises from 20.0% to 60.0% (Tab. 3). Under the combined observation-perturbation setting, it has the highest result on all three tasks. These results provide an initial real-robot validation of the design. The improvement is largest on the deformable-object task, while clean-condition Pick Banana is tied at 100.0% and Pull Drawer shows a smaller gain.

Across the simulation benchmarks and the real robot, the largest observed gains occur on the more demanding settings: the Hard and Very Hard MetaWorld tiers, LIBERO-Long, and *Fold Cloth*. LIBERO Object/Goal and clean-condition Pick Banana are near saturation, so the aggregate improvement is not a uniform shift across all task groups. This pattern motivates the controlled studies below, which examine the compact interface separately from the overall policy comparisons.

4.3 Ablations and Diagnostics

We organize the controlled studies around four questions: how much predictive bandwidth is needed, how the one-token target should be constructed, how the future-latent stream behaves, and whether the gains persist across action horizons. We first examine success and measured cost across token budgets, then training checkpoints and transfer across backbones. These studies evaluate the complete compression configurations; the remaining ablations examine target construction and representation quality. Each study is interpreted within its stated evaluation protocol.

4.3.1 Q1: How Much Predictive Bandwidth Is Needed?

We first investigate predictive bandwidth at an action horizon of $H = 5$. Here, K denotes the number of tokens retained per view and frame. Table 4 compares dense encoder features ($K = 256$) with compact configurations ($K \in \{1, 3, 6\}$). All compact variants use direct-K outputs: the three pooling branches remain fused, and K is the number of fused tokens passed to the model. The same compression is used for current observations and future targets, so this study compares complete compression configurations.

Inference feasibility and compute overhead. Preserving dense encoder features ($K = 256$) produces a recorded sequence of more than $2{,}500$ tokens and runs out of memory in the measured setup. All three compact configurations run within that memory budget. From $K = 1$ to $K = 6$, trainable parameters increase by 0.05 M, from 482.73 M to 482.78 M, while measured latency rises by 18.5%, from 44.69 ms to 52.95 ms. The table's sequence counts should be distinguished from the generated suffix alone, whose length is $H(2K + 1)$ for two views and one action token per step. Latents and actions are sampled synchronously, rather than through an autoregressive rollout.

Policy grounding performance. In the primary ablation (Tab. 4), the one-token configuration reaches 61.53% average success, compared with 60.63% and 59.43% for three and six tokens.

Table 5: Average success rates (%) across training checkpoints (horizon 5) on MetaWorld MT50. Bold marks each row's highest value.

Checkpoint	$K = 1$	$K = 3$	$K = 6$
10k	48.12	**48.72**	35.91
20k	**60.25**	49.37	58.92
30k	**61.53**	60.63	59.43
40k	58.10	58.07	**64.80**
50k	**68.31**	64.20	64.05
60k	**72.01**	62.03	61.49

Table 6: OneWM performance across VLA backbones on MetaWorld MT50 at horizon 5: average success rates (%) and paired gains.

Backbone	Steps	$K = 1$	$K = 3$	Difference
$\pi_{0.5}$	10k	**62.4**	60.0	+2.4
	20k	**65.4**	65.0	+0.4
	30k	**70.6**	68.2	+2.4
SmolVLA	10k	**66.2**	61.4	+4.8
	20k	**71.6**	68.2	+3.4
	30k	**74.8**	69.6	+5.2

Table 7: Pooling ablation on LIBERO at horizon 5 under the common 30k-step adaptation schedule.

Method	**Spatial**	**Object**	**Goal**	**Long**	**Avg%**
OneWM-VLA	**96.6**	**96.8**	**94.6**	**85.0**	**93.3**
Static avg-pooling	77.8	78.0	94.0	41.4	72.8
No branch fusion	72.4	70.4	43.0	4.4	47.6

Table 8: Pooling branch ablation on MetaWorld MT50 (at horizon 5, 30k steps).

Method	**Easy**	**Medium**	**Hard**	**V-Hard**	**Avg%**
OneWM-VLA	79.29	**41.82**	**65.00**	60.00	**61.53**
Only LEARN	78.57	29.09	35.00	60.00	50.67
Only MAX	53.21	16.36	0.00	20.00	22.39
Only SUM	**86.43**	26.36	46.67	**62.00**	55.36

Table 9: Adaptive pooling and sequence-to-one MLP targets across action horizons. Entries are MetaWorld average success rates; the last column averages the six horizons.

Constructor	$H = 5$	10	15	20	25	30	Avg. Success (%)
MLP	58.91	**52.36**	40.18	34.91	35.27	35.85	42.91
Adaptive pooling	**61.53**	52.18	**52.41**	**58.09**	**45.91**	**52.87**	**53.83**

Together with its lower measured latency, these results support the compact configuration as a practical operating point.

Training checkpoints. Table 5 compares $K \in \{1, 3, 6\}$ under the same training and evaluation settings. At 60k, success is 72.01%, 62.03%, and 61.49%, respectively: one token is higher by 9.98 points than the best wider configuration at this checkpoint. It is highest at four of the six reported checkpoints, with $K = 3$ higher at 10k and $K = 6$ higher at 40k. These results support the compact configuration at the longer training budget, while showing that its advantage varies during training.

Transfer across VLA backbones. We also integrate OneWM into $\pi_{0.5}$ and SmolVLA-0.45B. One token is higher than three in all six paired backbone/checkpoint comparisons (Tab. 6), with margins of 2.4 and 5.2 points at 30k on the two backbones. This extends the observed ordering beyond π_0, although the smallest margin is only 0.4 points. Both variants include OneWM, so these comparisons measure the relative performance of two compression configurations after integration.

4.3.2 Q2: How Should the Predictive Target Be Constructed?

We next examine how the encoder features are aggregated into predictive targets. The controlled studies compare the full adaptive multi-branch module with static pooling, removing fusion, and retaining one scoring branch. The LIBERO ablation uses a separate run with $H = 5$ and 30k training steps, so its absolute values should not be compared directly with the main checkpoint in Tab. 2. The full pooling module has the highest aggregate in both studies. On LIBERO, it reaches 93.3%, compared with 72.8% for static averaging, and removing branch fusion is substantially worse; the largest separation appears on Long (Tab. 7). On MetaWorld, the fused module reaches 61.53%, above every individual branch (Tab. 8). The SUM branch is stronger on two individual tiers, so the aggregate advantage does not imply that the fused module is best on every tier.

Target constructor at fixed bandwidth. We compare Adaptive Attention Pooling with a sequence-to-one MLP using identical encoder features and one-token, 2048-D targets, with the downstream flow, attention mask, training schedule, and evaluator held fixed. Adaptive pooling achieves higher

success at five of the six action horizons in Tab. 9, averaging 53.83% compared with 42.91% for the MLP, a gain of 10.92 percentage points. Over $H \in \{15, 20, 25, 30\}$, the mean gain is 15.77 points; the MLP is higher at $H = 10$ by 0.18 points. These results support Adaptive Attention Pooling as the target constructor across the tested action horizons.

4.3.3 Q3: What Does Future Supervision Contribute?

We first compare the full model with an architecture-matched model trained without the latent loss, using 30k training steps and $H = 20$ (Tab. 16). The architecture, training budget, horizon, camera settings, and evaluation protocol are identical; only the latent loss is removed. Average success falls from 58.09% to 21.64%, a 36.45-point decrease, with lower success in every difficulty group. This matched comparison supports the contribution of future-frame supervision to control.

We then examine the variation retained by the compact representation. Appendix F details the representation diagnostics reported in Tab. 17. At 30k steps, the $H = 5$ full model has a target entropy effective rank of 21.42 and a participation rank of 8.88, compared with 1.83 and 1.31 without the latent loss. The full model retains variation along multiple directions, while the no-loss control is close to rank one. We also compare the full-dimensional Fisher ratio before and after pooling on the same time-aligned LIBERO-Long samples (Appendix G). The pooled representation retains 77.3% of the reference Fisher ratio (Tab. 10), decreasing from 0.524 to 0.405. Compression therefore weakens task separation while preserving measurable between-task variation relative to within-task variation. This measures retained task variation, rather than the preservation of every control-relevant factor.

Table 10: Task separation on LIBERO-Long with normalized within-task scatter.

Representation	$\mathrm{tr}(S_B)$ **(norm.)**	$\mathrm{tr}(S_W)$ **(norm.)**	F
Before pooling	0.524	1.000	0.524
After pooling	0.405	1.000	0.405

4.3.4 Q4: Do the Gains Persist Across Action Horizons?

We next examine the non-primary horizons in Tab. 1. OneWM-VLA has higher aggregate success than π_0 at every matched horizon, but the margin is not monotonic. At the longest horizon, the gain is concentrated on Hard and Very Hard tasks, while π_0 remains stronger on Easy and Medium tasks. These results show that the observed advantages extend across the evaluated action-chunk lengths, without increasing monotonically with horizon.

5 Conclusion and Discussion

We presented OneWM-VLA, which represents each retained camera view with one predictive token per future step and generates predictive latents and actions through a joint conditional flow. This design introduces future-frame supervision without carrying dense visual features through the generative sequence. Across MetaWorld MT50, LIBERO, and a real Piper arm, it achieves higher aggregate success than π_0, with the clearest gains on difficult and extended interactions. The compact configuration has the highest success at the final checkpoint of the training comparison and outperforms the wider alternative in the paired studies on two additional VLA backbones. The ablations also highlight the roles of future-frame supervision, adaptive pooling, and future-target construction, while the representation analyses show nonconstant targets and retained task-related variation. Overall, our results support compact predictive latents as an effective interface between visual prediction and action generation in the evaluated settings.

Limitations. Our results support one token as an effective configuration rather than a universal optimum. The bandwidth sweep changes current-observation and future-target token counts together. The collapse diagnostics are empirical and provide no general guarantee, which would require additional assumptions or explicit objective and architectural constraints. The real-robot evaluation covers only a small set of tasks; broader studies across tasks and robots, and controls that isolate future-token count, remain future work.

AI USE STATEMENT

Generative AI tools were used for language editing and manuscript polishing. The authors reviewed all AI-assisted content and take full responsibility for the final manuscript.

ETHICS STATEMENT

We have identified no specific ethical concerns associated with this work.

REPRODUCIBILITY STATEMENT

The method and experimental protocols are described in Sections 3 and 4, with additional implementation and evaluation details provided in the appendix.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629. IEEE, 2023.

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

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

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, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2026. URL https://arxiv.org/abs/2410.24164.

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, Siteng Huang, Yuqian Yuan, Kehan Li, Hangjie Yuan, Chaohui Yu, Yuming Jiang, Jiayan Guo, Xin Li, Hao Luo, et al. Rynnvla-002: A unified vision-language-action and world model. *arXiv preprint arXiv:2511.17502*, 2025a.

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*, 2025b.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Xiang Li, Quanlu Zhang, Zhaofei Yu, et al. π_{RL}: Online rl fine-tuning for flow-based vision-language-action models. *arXiv preprint arXiv:2510.25889*, 2025.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

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.

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. *Advances in neural information processing systems*, 36:9156–9172, 2023.

Ankit Goyal, Hugo Hadfield, Xuning Yang, Valts Blukis, and Fabio Ramos. Vla-0: Building state-of-the-art vlas with zero modification. *arXiv preprint arXiv:2510.13054*, 2025.

Yanjiang Guo, Yucheng Hu, Jianke Zhang, Yen-Jen Wang, Xiaoyu Chen, Chaochao Lu, and Jianyu Chen. Prediction with action: Visual policy learning via joint denoising process. *Advances in Neural Information Processing Systems*, 37:112386–112410, 2024.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

Danijar Hafner, Wilson Yan, and Timothy Lillicrap. Training agents inside of scalable world models. *arXiv preprint arXiv:2509.24527*, 2025.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

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. *arXiv preprint arXiv:2412.14803*, 2024.

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. *arXiv preprint arXiv:2507.16815*, 2025.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, et al. Dreamgen: Unlocking generalization in robot learning through video world models. *arXiv preprint arXiv:2505.12705*, 2025.

Yuming Jiang, Siteng Huang, Shengke Xue, Yaxi Zhao, Jun Cen, Sicong Leng, Kehan Li, Jiayan Guo, Kexiang Wang, Mingxiu Chen, et al. Rynnvla-001: Using human demonstrations to improve robot manipulation. *arXiv preprint arXiv:2509.15212*, 2025.

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.

Yann LeCun. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

Jianxiong Li, Jinliang Zheng, Yinan Zheng, Liyuan Mao, Xiao Hu, Sijie Cheng, Haoyi Niu, Jihao Liu, Yu Liu, Jingjing Liu, et al. Decisionnce: Embodied multimodal representations via implicit preference learning. *arXiv preprint arXiv:2402.18137*, 2024a.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024b.

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. *arXiv preprint arXiv:2311.01378*, 2023.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023.

Dongxiu Liu, Haoyi Niu, Zhihao Wang, Jinliang Zheng, Yinan Zheng, Zhonghong Ou, Jianming Hu, Jianxiong Li, and Xianyuan Zhan. Efficient robotic policy learning via latent space backward planning. *arXiv preprint arXiv:2505.06861*, 2025.

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.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pp. 23301–23320. PMLR, 2023.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026.

Reginald McLean, Evangelos Chatzaroulas, Luc McCutcheon, Frank Röder, Tianhe Yu, Zhanpeng He, K.R. Zentner, Ryan Julian, J K Terry, Isaac Woungang, Nariman Farsad, and Pablo Samuel Castro. Meta-world+: An improved, standardized, RL benchmark. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025. URL https://openreview.net/forum?id=1de3azE606.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

Younggyo Seo, Danijar Hafner, Hao Liu, Fangchen Liu, Stephen James, Kimin Lee, and Pieter Abbeel. Masked world models for visual control. In *Conference on Robot Learning*, pp. 1332–1344. PMLR, 2023.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

Zuojin Tang, Xiaoyu Chen, Yongqiang Li, and Jianyu Chen. Efficient and generalized end-to-end autonomous driving system with latent deep reinforcement learning and demonstrations. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 179–197. Springer, 2025a.

Zuojin Tang, Bin Hu, Chenyang Zhao, De Ma, Gang Pan, and Bin Liu. Vlascd: A visual language action model for simultaneous chatting and decision making. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 9223–9243, 2025b.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Junjin Xiao, Yandan Yang, Xinyuan Chang, Ronghan Chen, Feng Xiong, Mu Xu, Wei-Shi Zheng, and Qing Zhang. World-env: Leveraging world model as a virtual environment for vla post-training. *arXiv preprint arXiv:2509.24948*, 2025.

Amber Xie, Oleh Rybkin, Dorsa Sadigh, and Chelsea Finn. Latent diffusion planning for imitation learning. *arXiv preprint arXiv:2504.16925*, 2025.

Angen Ye, Zeyu Zhang, Boyuan Wang, Xiaofeng Wang, Dapeng Zhang, and Zheng Zhu. Vla-r1: Enhancing reasoning in vision-language-action models. *arXiv preprint arXiv:2510.01623*, 2025.

Shuang Zeng, Dekang Qi, Xinyuan Chang, Feng Xiong, Shichao Xie, Xiaolong Wu, Shiyi Liang, Mu Xu, and Xing Wei. Janusvln: Decoupling semantics and spatiality with dual implicit memory for vision-language navigation. *arXiv preprint arXiv:2509.22548*, 2025.

Jianke Zhang, Yanjiang Guo, Yucheng Hu, Xiaoyu Chen, Xiang Zhu, and Jianyu Chen. Up-vla: A unified understanding and prediction model for embodied agent. *arXiv preprint arXiv:2501.18867*, 2025a.

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*, 2025b.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 1702–1713, 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.

Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 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 METAWORLD MT50 DIFFICULTY TIERS

Following Seo et al. (2023), we divide the 50 MetaWorld MT50 tasks into four tiers by manipulation difficulty (Tab. 11). We use this partition for every tier-level result in the main text. The checkpoint sweep in Tab. 5 uses camera $x = 0.75$ for all three token configurations.

Table 11: MetaWorld MT50 task-difficulty partition used in our evaluation, following Seo et al. (2023).

Difficulty	**Tasks**
Easy (28)	`button-press`, `button-press-topdown`, `button-press-topdown-wall`, `button-press-wall`, `coffee-button`, `dial-turn`, `door-close`, `door-lock`, `door-open`, `door-unlock`, `drawer-close`, `drawer-open`, `faucet-close`, `faucet-open`, `handle-press`, `handle-press-side`, `handle-pull`, `handle-pull-side`, `lever-pull`, `plate-slide`, `plate-slide-back`, `plate-slide-back-side`, `plate-slide-side`, `reach`, `reach-wall`, `window-close`, `window-open`, `peg-unplug-side`
Medium (11)	`basketball`, `bin-picking`, `box-close`, `coffee-pull`, `coffee-push`, `hammer`, `peg-insert-side`, `push-wall`, `soccer`, `sweep`, `sweep-into`
Hard (6)	`assembly`, `hand-insert`, `pick-out-of-hole`, `pick-place`, `push`, `push-back`
Very Hard (5)	`shelf-place`, `disassemble`, `stick-pull`, `stick-push`, `pick-place-wall`

B SENSITIVITY TO THE FUSION TEMPERATURE

We vary the pooling temperature τ_p used for token scoring and branch fusion (Sec. 3.1). Among the three tested values, τ_p=0.1 gives the highest average LIBERO success (Tab. 12). We therefore use it in the main experiments.

Table 12: LIBERO success rates (%) for different pooling temperatures τ_p.

τ_p	Spatial	Object	Goal	Long	Avg%
1.0	96.6	**97.8**	89.2	77.4	90.3
0.5	**97.4**	97.2	87.0	78.6	90.1
0.1	96.6	96.8	**94.6**	**85.0**	**93.3**

C FLOW-MATCHING DISTANCE METRIC

We compare L_1 and L_2 distances for velocity regression in the action and latent branches, holding all other settings fixed (Tab. 13). The L_1/L_1 configuration gives the highest average success among the three variants. We use it in the main experiments as an empirical velocity-regression choice; it does not inherit the conditional-mean guarantee associated with squared-error flow matching.

Table 13: LIBERO success rates (%) for different distance metrics in the action and latent flow-matching losses. Training and inference both use H=5.

Distance (Action / Latent)	Spatial	Object	Goal	Long	Avg%
L_2 / L_2	94.8	94.6	87.0	74.2	87.7
L_1 / L_2	96.4	**98.0**	88.2	78.2	90.2
L_1 / L_1	**96.6**	96.8	**94.6**	**85.0**	**93.3**

D LIBERO INFERENCE-TIME CONFIGURATION

We use one checkpoint across all four LIBERO suites and select the inference action horizon and replan step on a held-out validation subset. Tab. 14 lists the fixed configurations used for evaluation. We also sweep these two settings on LIBERO-Long with the same checkpoint (Tab. 15). The gray row is the reference configuration of the same checkpoint, and the blue row is the highest-scoring setting in the sweep. The $20/16$ setting already exceeds the 92.4% result of $\pi_{0.5}$, while $20/20$ is the longest inference horizon. Success ranges from 93.4% to 95.6% across the tested grid, while the representation stays fixed at one token per view and frame. For each suite, the selected pair is fixed for all evaluation rollouts and is not adapted per episode.

Table 14: Inference-time configuration of OneWM-VLA on LIBERO. All suites use the checkpoint trained with an action horizon of 20; the inference horizon and replan step are selected per suite from a small validation grid.

Suite	Inference horizon	Replan step	Avg. Success (%)
Spatial	14	10	98.2
Object	14	10	99.6
Goal	18	7	99.0
Long	18	12	95.6

E LATENT-LOSS ABLATION ON METAWORLD MT50

Table 16 compares the full OneWM-VLA model and its architecture-matched no-latent-loss control after 30k training steps, with $H = 20$ at training and inference. Both use the same camera settings and evaluation protocol from Sec. 4.1. The control retains the latent tokens, projections, attention mask, and action objective; only the latent-loss weight is set to zero. The full model reaches 58.09% average success, compared with 21.64% for the control, a difference of 36.45 percentage points.

F REPRESENTATION-DIAGNOSTIC PROTOCOL

Table 17 reports the target-spectrum and matching diagnostics. Both effective ranks are computed from the covariance of future-frame target vectors.

Checkpoints and samples. We evaluate the full one-token model and its architecture-matched no-latent-loss control, both trained with $H = 5$ for 30k steps. We use valid base-view records from 49 MetaWorld tasks and 245 episodes, yielding 1,960 contexts. Five future steps per context produce M_z=9,800 target vectors. Both models use the same records and covariance calculation.

Spectral metrics. Let $\widetilde{Z} \in \mathbb{R}^{M_z \times D}$ contain the centered target vectors. We compute

$$\begin{aligned} C_z &= \frac{1}{M_z - 1} \widetilde{Z}^\top \widetilde{Z}, \qquad p_j = \frac{\lambda_j}{\sum_k \lambda_k}, \\ r_{\text{ent}} &= e^{-\sum_j p_j \log p_j}, \quad r_{\text{part}} = \frac{1}{\sum_j p_j^2}, \end{aligned} \tag{7}$$

where $\{\lambda_j\}$ are the eigenvalues of C_z. Zero covariance corresponds to a constant representation; for nonzero covariance, both effective ranks approach one as the variance concentrates in a single direction.

Matching diagnostics. Queries are the generated 2048-D base-view latents for the five consecutive future frames $t + 1, \ldots, t + 5$, jointly sampled with actions conditioned on the current image, robot state, and language instruction under a common Gaussian initialization. For R@1, we subtract the corresponding global or within-task target mean from both queries and pooled future-frame targets, apply ℓ_2 normalization, and rank targets at the same future offset by cosine similarity. The

Table 15: LIBERO-Long success rates (%) across inference action horizons and replan steps using a common checkpoint trained with an action horizon of 20.

Train/Infer.	Replan	Suffix tokens	Avg. Success (%)
20/15 (baseline)	10	45	93.4
20/16	10	48	94.0
20/18	10	54	94.8
20/18	11	54	95.4
20/18	12	54	**95.6**
20/20	10	60	94.0

Table 16: Latent-loss ablation on MetaWorld MT50 ($H = 20$, 30k training steps). Δ_{full} is the percentage-point difference from the full model.

Method	**Easy**	**Med.**	**Hard**	**V-Hard**	**Avg%**	Δ_{full}
OneWM-VLA	**79.64**	**42.73**	**60.00**	**50.00**	**58.09**	–
No latent loss	45.36	14.55	6.67	20.00	21.64	-36.45

Table 17: Representation diagnostics for the full one-token model and its architecture-matched no-latent-loss control, both trained with $H = 5$ for 30k steps. The last row reports the separate architecture-matched policy evaluation at $H = 20$ after 30k training steps.

Diagnostic	**Full one-token, 30k**	**No latent loss, 30k**
Target entropy effective rank	**21.42**	1.83
Covariance participation rank	**8.88**	1.31
Temporal discrimination	**77.09%**	50.04%
Cross-task discrimination	**73.52%**	49.18%
Global R@1	**0.510%**	0.051%
Global chance	0.051%	0.051%
Within-task R@1	**7.53%**	2.43%
Within-task chance	2.50%	2.50%
Policy avg. success ($H = 20$)	**58.09%**	21.64%

candidate sets contain $1{,}960$ global targets or 40 same-task targets, giving chance levels of 0.051% and 2.50%, respectively. Pairwise discrimination measures how often the matched target has a lower target-variance-normalized MSE than a negative, with a 50% chance level. Temporal negatives are constructed by cyclically shifting the eight anchors within each episode by four anchor positions; cross-task negatives are constructed by rotating task identities while preserving the episode slot, anchor slot, and future offset.

Interpretation. The full model has target variance distributed across multiple directions, while the no-loss control is close to rank one. The separate $H = 20$ policy evaluation at 30k also favors the full model by 36.45 percentage points (Tab. 16). The spectral measurements describe representation variation, while the policy evaluation measures the effect of removing the latent loss on task success.

G Feature-Space Separation Analysis

Features. We extract visual features only, without language or action tokens. Before pooling, each sample contains the 256 visual tokens produced by the PaliGemma encoder. We average these tokens to obtain one feature vector per sample, so the reference is mean-pooled encoder features rather than the full patch sequence. After pooling, each sample is represented directly by the single output token of Adaptive Attention Pooling. This construction gives one full-dimensional feature vector per sample in both settings.

Sampling. We use 24 successful trajectories from each of the 10 LIBERO-Long tasks. One frame is extracted from each trajectory at the same relative time, yielding 240 samples in total. The same samples are used before and after pooling.

Fisher ratio. Let $\mathcal{I}_c$ contain the samples from task c, let $M_c = |\mathcal{I}_c|$, and let $\boldsymbol{\mu}_c$ and $\boldsymbol{\mu}$ denote the task and global feature means. We compute task separation in the original feature space as

$$F = \frac{\sum_c M_c \, \|\boldsymbol{\mu}_c - \boldsymbol{\mu}\|_2^2}{\sum_c \sum_{n \in \mathcal{I}_c} \|\mathbf{f}_n - \boldsymbol{\mu}_c\|_2^2} = \frac{\mathrm{tr}(S_B)}{\mathrm{tr}(S_W)}, \tag{8}$$

where S_B and S_W are the between-task and within-task scatter matrices. The metric is computed from the full feature vectors rather than a low-dimensional projection.

Because the two representations have different feature scales, we normalize both scatter traces by $\mathrm{tr}(S_W)$ within each setting when reporting Tab. 10. This sets the normalized within-task trace to one and leaves the Fisher ratio unchanged. The resulting comparison measures the relative change in task separation, rather than differences in raw feature magnitude.