STaC-WAM: Asymmetric World-Action Models via Spatio-Temporal Action-Conditioning

Anonymous authors
Paper under double-blind review

Abstract

World Action Models (WAMs) improve robot policies by coupling action generation with predictive visual modeling. However, efficient inference remains challenging. Implicit WAMs avoid test-time future-video generation by using a predictive video backbone, but they still lack an explicit mechanism for transforming predictive visual representations into compact action-conditioning features. Video features encode high-dimensional visual dynamics, including object interactions, motion patterns, and local details, whereas action generation only requires a subset relevant for control. Without such an interface, the action expert must implicitly learn this transformation through extra capacity, making action generation computationally expensive. We introduce STaC-WAM, an asymmetric implicit WAM that couples a large predictive video backbone with a lightweight action expert through a **Spatio-Temporal Action-Conditioning (STaC) bridge.** Instead of directly exposing the action expert to deep video representations, STaC compresses multi-layer video features into compact action-conditioning representations while preserving object-level structure and fine-grained visual details. Our experiments suggest a broader design principle for WAMs with predictive visual backbones: visual modeling and action generation need not scale symmetrically. With the STaC bridge, a 6-layer Action DiT outperforms the 30-layer Fast-WAM baseline on LIBERO, improving success rate from 97.6% to 98.85% while increasing inference speed by 2.10x, with pronounced gains on LIBERO-Long. Evaluations on simulated and real-world tasks further demonstrate the robustness of this asymmetric design. Ablations show that Temporal Similarity Alignment provides the largest gain, while Object Assignment Alignment and the Object-Guided Dual-Path Decoder provide complementary benefits under aggressive compression.

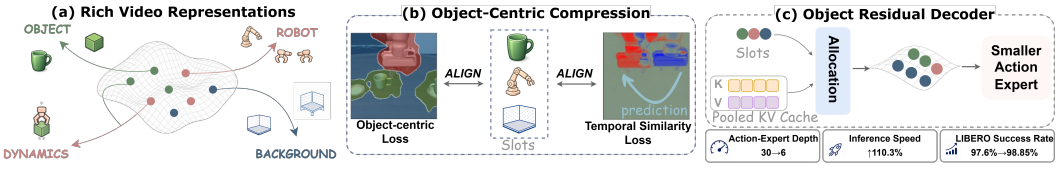


Figure 1: Motivation for STaC-WAM. (a) **Rich Video Representations** contain diverse visual information, while action generation requires only compact task-relevant cues. (b) **Object-Centric Compression** distills predictive features into structured slots using Object Assignment Alignment and Temporal Similarity Alignment. (c) **Object-Guided Dual-Path Decoder** combines object-level slot features with complementary pooled KV features, enabling a smaller action expert and faster inference without performance loss.

1 Introduction

General robot policies require both dynamic visual understanding and precise action generation. World-action models (WAMs) address this challenge by coupling action generation with predictive visual modeling. Existing WAMs can be broadly divided into explicit approaches, which generate future visual states at test time, and implicit approaches, which use the video backbone as a predictive feature extractor. Although implicit WAMs avoid test-time future generation, they still face a

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.

mismatch between rich hierarchical predictive representations and the compact information required for action generation. As illustrated in Figure 1, predictive features encode object configurations, motion, interactions, and appearance details, while control requires only a subset of this information. Without a structured interface, the action expert must learn this transformation together with action generation, increasing capacity and inference cost. A desirable interface should therefore preserve control-relevant information while removing redundant visual variation.

We introduce STaC-WAM, an asymmetric implicit WAM that enables independent scaling of predictive visual modeling and action generation through a Spatio-Temporal Action-Conditioning (STaC) bridge. Rather than directly exposing the action expert to deep layer-wise video representations, STaC transforms multi-layer predictive features into compact conditioning representations while preserving manipulation-relevant information. Specifically, STaC employs an object-centric spatio-temporal compressor with Object Assignment Alignment (OAA) and Temporal Similarity Alignment (TSA) to preserve object organization and short-horizon dynamics. We further introduce an Object-Guided Dual-Path Decoder, which combines object-level slot representations with complementary pooled visual features and uses object-guided routing for dense feature reconstruction.

We evaluate STaC-WAM on LIBERO, RoboTwin, and real-robot manipulation tasks. In our main implementation, STaC maps each group of five video-backbone layers to one action-conditioning layer, reducing the interface depth from 30 stages to 6. On LIBERO, a 6-layer Action DiT with STaC achieves 98.85% average success rate over 2000 evaluation trials, outperforming the original 30-layer Fast-WAM baseline. Results on RoboTwin and real-robot tasks further demonstrate strong performance with faster inference. These results show that predictive visual modeling and action generation can be scaled asymmetrically through an effective video-to-action interface.

Our contributions are:

- We reveal that implicit WAMs do not require symmetric scaling between predictive visual models and action experts when an appropriate video-to-action interface is introduced, and formulate the asymmetric scaling problem for efficient WAM design.
- We introduce STaC, a structured video-to-action conditioning bridge that transforms hierarchical predictive representations into compact action-conditioning features, enabling a lightweight action expert without directly inheriting the depth of the predictive model.
- We develop an object-centric compression and reconstruction framework with structure-preserving alignment objectives and an Object-Guided Dual-Path Decoder, which preserves object structures and fine-grained details during aggressive representation compression.

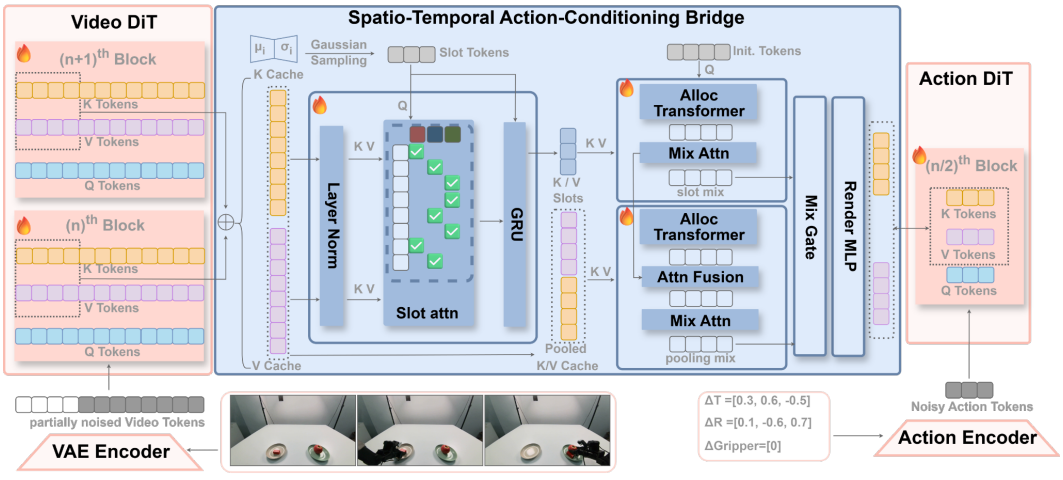


Figure 2: Overview of STaC-WAM. The predictive world expert extracts hierarchical representations. The proposed STaC bridge compresses these representations through object-centric slots and reconstructs dense action-conditioning features with the Object-Guided Dual-Path Decoder before action generation.

2 Related Work

2.1 World-Action Models for Robot Policies

Vision-language-action models map observations and language instructions to robot actions using autoregressive, diffusion, or flow-based decoders (Kim et al., 2025b; Black et al., 2025b; Chi et al., 2024; Li et al., 2025c; 2026c; Hu et al., 2026a; Li et al., 2025b; Yue et al., 2024; Zhang et al., 2026a; Ni et al., 2026; Reuss et al., 2025; Hu et al., 2026b;c). World-action models (WAMs) extend this paradigm by coupling action generation with predictive visual modeling, allowing robot policies to leverage visual-dynamics priors learned from large-scale video data. Existing WAMs can be categorized into explicit and implicit approaches. Explicit WAMs generate future visual states during inference and use predicted observations or latent states for action generation (Bi et al., 2026; Ye et al., 2026b; Kim et al., 2026), but introduce additional inference overhead. In contrast, implicit WAMs avoid test-time future-video generation by directly using the video backbone as a predictive feature extractor (Yuan et al., 2026; Ma et al., 2026; Zhang et al., 2026b; Chen et al., 2026a). However, they still expose action experts to deep and high-dimensional predictive representations, leaving the video-to-action conditioning problem unresolved.

2.2 Efficient Robot Policy Inference

Recent efforts on efficient WAMs mainly focus on reducing the computational burden of predictive visual modeling. Fast-WAM avoids expensive test-time future-video generation by directly using predictive representations for action generation, while subsequent approaches improve efficiency through lightweight video models, compact visual representations, or more efficient world-model inference strategies (Yuan et al., 2026; Li et al., 2026a;d; Ye et al., 2026a).

Another line of work accelerates robot policy inference by reducing action generation cost, such as decreasing iterative sampling steps in diffusion or flow-based policies (Chen et al., 2026b). However, these approaches mainly optimize visual dynamics modeling or action decoding, while we focus on the underexplored video-to-action interface: transforming rich predictive representations into compact action-conditioning features for lightweight action experts.

2.3 Object-Centric Representations for Embodied Learning

Object-centric representation learning aims to discover compact and structured representations from dense visual observations. Unlike dense representations that entangle objects, backgrounds, and appearance variations, object-centric methods organize visual information into entity-level latent components. Slot Attention and its extensions learn object-like decompositions through iterative attention mechanisms, enabling reusable latent representations (Locatello et al., 2020). Recent approaches further incorporate pretrained visual representations, reconstruction objectives, and temporal consistency constraints to extend object-centric learning to complex visual environments (Seitzer et al., 2023; Sajjadi et al., 2022; Zadaianchuk et al., 2023).

Such structured representations are useful for embodied learning, where actions depend on object states and interactions. Unlike prior work on structured visual representation learning, we use object-centric representations as an interface that transforms predictive world features into compact action-conditioning features.

3 Method

3.1 Implicit WAM Formulation

The overall architecture of STaC-WAM is shown in Figure 2. A world-action model (WAM) couples predictive visual modeling with action generation. Explicit WAMs generate future visual states during inference, whereas implicit WAMs directly leverage predictive world representations to condition action generation.

An implicit WAM consists of both a world expert and an action expert:

$$z = E_w(o, l), \qquad a = E_a(z, l), \tag{1}$$

The world expert produces hierarchical key–value feature representations:

$$\mathcal{F} = \left\{ F_i = (F_i^K, F_i^V) \right\}_{i=1}^{N}, \tag{2}$$

where F_i^K and F_i^V are generated by the i-th world block. Instead of directly coupling all world representations with an equally deep action expert, we introduce a compact action-conditioning interface to transform the predictive world features into efficient action conditions.

3.2 Spatio-Temporal Action-Conditioning Bridge

We introduce the Spatio-Temporal Action-Conditioning (STaC) bridge between the world and action experts. STaC groups adjacent world representations and converts each group into a dense action-conditioning representation.

For the g-th group, the input representations are formulated as below:

$$X_g^\xi = [F_{r(g-1)+1}^\xi; \ldots; F_{rg}^\xi], \qquad \xi \in \{K, V\}, \tag{3}$$

where r denotes the number of grouped world layers, $G = N/r$ is the number of resulting groups, and $[\cdot;\cdot]$ represents token-wise concatenation. Each group is transformed by:

$$\widetilde{F}_g^\xi = \mathrm{STaC}^\xi \left(X_g^\xi \right), \qquad g = 1, \ldots, G. \tag{4}$$

The grouping factor r controls the compression ratio of the interface. Within each group, STaC performs object-centric compression, structure-preserving alignment, and feature reconstruction through an Object-Guided Dual-Path Decoder.

3.3 Object-Centric Compression and Alignment

Given grouped world features, Slot Attention compresses tokens into M latent slots with token-to-slot assignments:

$$\left(S_g^\xi, A_g^{\mathrm{cmp},\xi} \right) = \mathrm{SlotAttn}^\xi \left(X_g^\xi \right), \tag{5}$$

where $S_g^\xi \in \mathbb{R}^{M \times d}$ denotes the compressed slots and $A_g^{\mathrm{cmp},\xi}$ denotes the corresponding assignment maps. The assignments are organized according to the r source layers:

$$A_g^{\mathrm{cmp},\xi} = [A_{g,1}^{\mathrm{cmp},\xi}; \ldots; A_{g,r}^{\mathrm{cmp},\xi}], \tag{6}$$

where $A_{g,j}^{\mathrm{cmp},\xi} \in \mathbb{R}^{L \times M}$.

To preserve manipulation-relevant information during compression, we introduce two training objectives: Object Assignment Alignment (OAA) and Temporal Similarity Alignment (TSA).

Object Assignment Alignment. OAA encourages slot assignments to preserve object-level spatial organization. The object-assignment targets are obtained from a separately trained VideoSAUR model using self-supervised video representation learning, without requiring manual object annotations. For each training sample, VideoSAUR produces an object-centric assignment map, which is mapped to the corresponding visual-token grid. We denote the resulting target for the j-th source layer in group g as $Y_{g,j} \in \mathbb{R}^{L \times M}$, where each row specifies the assignment distribution of a visual token over the M object slots. These targets supervise the slot-assignment maps $A_{g,j}^{\mathrm{cmp},\xi}$ produced by STaC. Since slot identities are unordered, we perform permutation-invariant matching over the slot dimension:

$$\ell_{\mathrm{oaa}} = \frac{1}{G} \sum_{g=1}^{G} \min_{\pi \in \mathfrak{S}_M} \frac{1}{2r} \sum_{\xi \in \{K,V\}} \sum_{j=1}^{r} \mathrm{CE} \left(Y_{g,j,\pi}, A_{g,j}^{\mathrm{cmp},\xi} \right), \tag{7}$$

where $\mathfrak{S}_M$ denotes the set of permutations of the M slots, and $Y_{g,j,\pi}$ denotes $Y_{g,j}$ with its slot dimension permuted according to π. VideoSAUR is used only to generate training targets and is not part of the STaC inference pipeline.

Temporal Similarity Alignment. TSA preserves short-horizon correspondence patterns from the original predictive representations. We aggregate features within each group:

$$\bar{F}^{\xi}_{g,t} = \frac{1}{r}\sum_{j=1}^{r} F^{\xi}_{r(g-1)+j,t}, \tag{8}$$

and normalize them as:

$$U^{\xi}_{g,t} = \mathrm{L2Norm}\left(\bar{F}^{\xi}_{g,t}\right). \tag{9}$$

The source temporal similarity distribution is defined as:

$$P^{\mathrm{src},\xi}_{g,t} = \mathrm{sg}\left[\mathrm{softmax}\left(\frac{U^{\xi}_{g,t}(U^{\xi}_{g,t+1})^{T}}{\tau}\right)\right], \tag{10}$$

where $\mathrm{sg}[\cdot]$ denotes stop-gradient and τ is the temperature parameter.

A lightweight prediction head estimates the similarity distribution from reconstructed conditioning features:

$$P^{\mathrm{pred},\xi}_{g,t} = \mathrm{softmax}\left(\mathcal{H}^{\xi}_{\mathrm{tsa}}(\widetilde{F}^{\xi}_{g,t})\right). \tag{11}$$

The TSA objective is:

$$\ell_{\mathrm{tsa}} = \frac{1}{2G}\sum_{g=1}^{G}\sum_{\xi\in\{K,V\}} \mathrm{CE}\left(P^{\mathrm{src},\xi}_{g,t}, P^{\mathrm{pred},\xi}_{g,t}\right). \tag{12}$$

3.4 Object-Guided Dual-Path Decoder

Although object-centric slots provide compact structural representations, aggressive compression may remove fine-grained visual information required for precise manipulation. We therefore reconstruct dense conditioning features through two complementary pathways: an object pathway that preserves entity-level structure and a detail pathway that recovers complementary spatial information from the original world representations.

The object pathway decodes the compressed slots into position-aligned features:

$$\left(H^{\mathrm{obj},\xi}_{g}, A^{\mathrm{obj},\xi}_{g}\right) = \mathcal{D}^{\mathrm{obj},\xi}\left(S^{\xi}_{g}, P^{\xi}\right), \tag{13}$$

where P^{ξ} denotes learned positional embeddings and $A^{\mathrm{obj},\xi}_{g} \in \mathbb{R}^{L\times M}$ associates reconstructed positions with the compressed slots.

The detail pathway preserves complementary visual information from the original grouped features. We first aggregate the source features and slot assignments:

$$\bar{F}^{\xi}_{g} = \frac{1}{r}\sum_{j=1}^{r} F^{\xi}_{r(g-1)+j}, \qquad \bar{A}^{\mathrm{cmp},\xi}_{g} = \frac{1}{r}\sum_{j=1}^{r} A^{\mathrm{cmp},\xi}_{g,j}. \tag{14}$$

Using the shared slot correspondence between the reconstructed and source representations, we construct a slot-mediated spatial prior:

$$\Pi^{\xi}_{g} = \mathrm{RowNorm}\left(A^{\mathrm{obj},\xi}_{g}\left(\bar{A}^{\mathrm{cmp},\xi}_{g}\right)^{T}\right), \tag{15}$$

where $\Pi^{\xi}_{g} \in \mathbb{R}^{L\times L}$ transfers spatial correspondence from the original world features to the reconstructed positions.

The detail features are reconstructed as:

$$H^{\mathrm{det},\xi}_{g} = \mathcal{D}^{\mathrm{det},\xi}\left(\bar{F}^{\xi}_{g}, \Pi^{\xi}_{g}, P^{\xi}\right). \tag{16}$$

Finally, a token-wise router dynamically combines the two complementary pathways:

$$[w^{\mathrm{obj},\xi}_{g}, w^{\mathrm{det},\xi}_{g}] = \mathrm{Softmax}_2\left(\mathcal{R}^{\xi}\left(H^{\mathrm{obj},\xi}_{g}, H^{\mathrm{det},\xi}_{g}, \bar{F}^{\xi}_{g}\right)\right), \tag{17}$$

$$H^{\mathrm{fuse},\xi}_{g} = w^{\mathrm{obj},\xi}_{g} \odot H^{\mathrm{obj},\xi}_{g} + w^{\mathrm{det},\xi}_{g} \odot H^{\mathrm{det},\xi}_{g}. \tag{18}$$

The fused features are rendered into the final action-conditioning representation:

$$\widetilde{F}^{\xi}_{g} = \mathrm{Render}^{\xi}\left(H^{\mathrm{fuse},\xi}_{g} + P^{\xi}\right). \tag{19}$$

3.5 Training Objective

The complete training objective combines the original WAM objectives with the proposed alignment losses:

$$\mathcal{L} = \lambda_{\text{vid}}\mathcal{L}_{\text{vid}} + \mathcal{L}_{\text{act}} + \lambda_{\text{oaa}}\ell_{\text{oaa}} + \lambda_{\text{tsa}}\ell_{\text{tsa}}, \tag{20}$$

where $\mathcal{L}_{\text{vid}}$ and $\mathcal{L}_{\text{act}}$ denote the original objectives. As illustrated in Figure 3, OAA and TSA preserve complementary structural and temporal information without overhead.

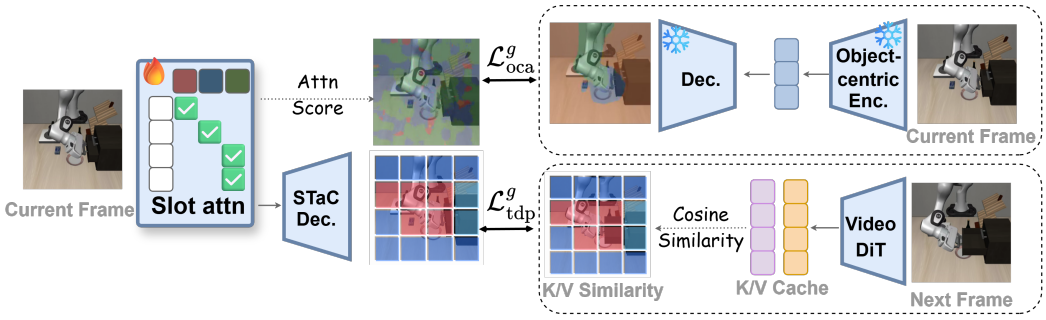


Figure 3: Structure-preserving alignment objectives in STaC. Object Assignment Alignment preserves object-level organization, while Temporal Similarity Alignment maintains short-horizon token correspondences.

3.6 Action Conditioning Interface

The reconstructed features form the compressed world cache:

$$\widetilde{K}_g = \widetilde{F}^K_g, \qquad \widetilde{V}_g = \widetilde{F}^V_g, \qquad g = 1, \ldots, G. \tag{21}$$

Each reconstructed key–value group conditions one Action DiT block:

$$H^{\text{act}}_g = E_{a,g}\left(H^{\text{act}}_{g-1}; \widetilde{K}_g, \widetilde{V}_g\right). \tag{22}$$

Here, $E_{a,g}$ preserves the original per-block Action DiT design. Therefore, STaC transforms the original N layer-wise world representations into G compact conditioning groups, reducing the required action expert depth from N to G while maintaining the original world-conditioned action-generation mechanism.

4 Experiments

4.1 Experimental Setup

We evaluate STaC-WAM on LIBERO, RoboTwin, and real-robot manipulation tasks. LIBERO contains four task suites: LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long. Following previous work, we evaluate 50 episodes per task, resulting in 2000 evaluation trials.

We compare STaC-WAM with representative policy baselines, including Diffusion Policy, Vision-Language-Action models, and World-Action Models. Fast-WAM serves as the primary controlled comparison: both methods use the same video backbone, observations, language instructions, robot-state inputs, and evaluation protocol. Fast-WAM uses a 30-layer Action DiT that consumes layer-wise video representations, while STaC-WAM replaces this depth-coupled interface with STaC and enables a shallower action expert.

We report task success rate and inference latency. Our models are trained and evaluated for task performance on NVIDIA H200 GPUs. Efficiency measurements, including inference latency, are conducted on an NVIDIA RTX 5090D GPU with batch size 1 under identical inference settings.

Table 1: Comparison of robot policies on the LIBERO benchmark. Success rates (%) are reported on four task suites: LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long. Bold values indicate the best performance in each column.

Method	Spatial	Object	Goal	Long	Avg.
Policy Baselines					
Diffusion Policy (Chi et al., 2024)	78.3	92.5	68.3	50.5	72.40
Star (Li et al., 2025a)	95.5	98.3	95.0	88.5	94.30
Vision–Language–Action Models					
OpenVLA (Kim et al., 2025b)	84.7	88.4	79.2	53.7	76.50
SpatialVLA (Qu et al., 2025b)	88.2	89.9	78.6	55.5	78.10
OpenVLA-OFT (Kim et al., 2025a)	97.6	98.4	97.9	94.5	97.10
π_0 (Black et al., 2025b)	96.8	98.8	95.8	85.2	94.15
$\pi_{0.5}$ (Black et al., 2025a)	98.8	98.2	98.0	92.4	96.85
GR00T-N1 (Bjorck et al., 2025; Qu et al., 2025a)	94.4	97.6	93.0	90.6	93.90
CogVLA (Li et al., 2025c)	98.6	98.8	96.6	95.4	97.40
BehaviorVLA (Hu et al., 2026a)	99.2	99.4	**98.8**	94.6	98.00
World–Action Models					
Motus (Bi et al., 2026; Chen et al., 2026a)	96.8	99.8	96.6	97.6	97.70
Cosmos Policy (Kim et al., 2026)	98.1	**100.0**	98.2	97.6	98.50
LingBot-VA (Li et al., 2026b)	98.5	99.6	97.2	**98.5**	98.45
Fast-WAM (Yuan et al., 2026)	98.2	**100.0**	97.0	95.2	97.60
STaC-WAM	**99.6**	99.4	98.4	98.0	**98.85**

Table 2: Performance comparison on the full RoboTwin benchmark. We report success rates (%) under both clean and randomized settings with 100 trials per setting. Eight representative tasks are shown for compact presentation; the average is computed over the full benchmark. Complete per-task results are given in Table 4 in the Appendix. Best results are highlighted in bold.

Task	Fast-WAM (Yuan et al., 2026)		Fast-WAM-Joint (Yuan et al., 2026)		Fast-WAM-IDM (Yuan et al., 2026)		LingBot-VA (Li et al., 2026b)		$\pi_{0.5}$ (Black et al., 2025a)		Motus (Bi et al., 2026)		STaC-WAM	
	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.
Handover Mic	99	**100**	**100**	**100**	98	99	94	96	98	97	78	63	**100.0**	**100.0**
Stack Blocks Two	**100**	**100**	**100**	**100**	**100**	**100**	**100**	98	97	**100**	**100**	98	**100.0**	**100.0**
Stack Bowls Two	92	**98**	97	95	94	96	94	**98**	95	96	**98**	**98**	97.0	97.0
Place Container Plate	96	**100**	99	98	**100**	96	99	97	99	95	98	99	99.0	99.0
...	...		...		...		...		...		...		...	
Place Object Basket	89	88	86	81	87	82	**91**	88	80	76	81	87	90.0	**90.0**
Place Object Stand	90	94	92	98	96	**100**	**99**	96	91	85	98	97	94.0	94.0
Place Burger Fries	96	99	**100**	**100**	97	99	97	95	94	87	98	98	98.5	98.5
Place Bread Basket	91	93	89	94	91	**97**	**97**	95	77	64	91	94	93.0	93.0
Average	91.88	91.78	90.84	90.32	91.16	91.34	**92.90**	91.50	82.74	76.76	88.66	87.02	92.25	**92.11**

4.2 Main Results

4.2.1 LIBERO Evaluation

Table 1 reports the main LIBERO results. STaC-WAM achieves a 98.85% average success rate with a 6-layer Action DiT, outperforming the 30-layer Fast-WAM baseline while achieving 2.10× faster inference. The gain is particularly notable on LIBERO-Long, where success increases from 95.2% to 98.0%, indicating that the compressed interface preserves predictive visual information required for manipulation.

4.2.2 RoboTwin Evaluation

We further evaluate STaC-WAM on the full RoboTwin benchmark. Each task is evaluated under clean and randomized settings with 100 trials per setting. Table 2 reports eight representative tasks and the full-benchmark average under the same protocol; complete per-task results are provided in Table 4 in the Appendix. STaC-WAM keeps the predictive video backbone unchanged while the action expert operates on compressed conditioning features, and maintains strong performance in both settings.

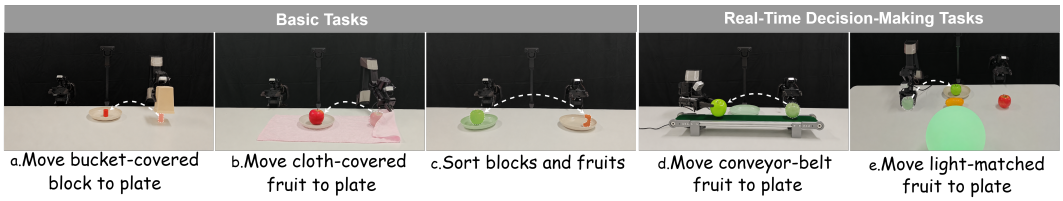
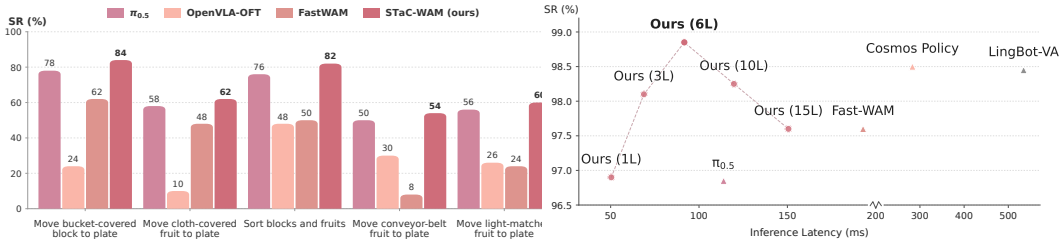


Figure 4: Real-world experimental setup illustrating the basic manipulation tasks and real-time decision-making tasks.



(a) Real-robot success rates

(b) Performance–efficiency trade-off

Figure 5: **Real-robot performance and inference efficiency.** (a) Success-rate comparison of different policies across real-robot manipulation tasks. (b) Performance–efficiency trade-off under different action-conditioning depths. Each point denotes a different Action DiT depth (L).

4.2.3 Real-Robot Evaluation

We further evaluate STaC-WAM on real-robot tasks involving object rearrangement, occlusion handling, visual attribute understanding, and dynamic object interaction. The tasks are shown in Figure 4, and success rates are summarized in Figure 5(a). Fast-WAM and STaC-WAM use identical observations and action-generation settings.

4.3 Efficiency Analysis

STaC-WAM reduces inference cost by decoupling the depth of the predictive visual model from the action expert. STaC compresses the video-action interface once per observation, allowing a smaller Action DiT to be reused throughout denoising and reducing end-to-end latency from 192.83 ms to 91.68 ms. The speedup does not come from fewer parameters: STaC-WAM contains 6.59B parameters, including the 1.38B-parameter STaC bridge, compared with 6.02B for Fast-WAM. Instead, computation is shifted away from the repeatedly executed action expert. In a corresponding 91.44 ms profiling configuration, STaC compression accounts for approximately 14–18 ms and the five-step 6-layer Action DiT for roughly 43 ms; these values are reconstructed from component profiling and depth-sweep measurements.

4.4 Ablation Studies

Unless otherwise specified, component ablations are conducted on LIBERO using a fixed 6-layer Action DiT and the same STaC architecture. The conditioning-depth ablation explicitly varies the Action DiT depth.

4.4.1 Conditioning Depth

We study how the number of action-conditioning layers affects task performance and inference efficiency. As shown in Figure 5(b), reducing depth improves efficiency and performance until 6 layers, while excessive compression removes predictive information and degrades manipulation performance.

4.4.2 COMPONENTS OF STAC

We further analyze the contribution of different components in STaC, including the compression strategy, alignment objectives, and the Object-Guided Dual-Path Decoder.

Table 3: **Component ablations of STaC.** Left: compression strategies and alignment objectives. Right: variants of the Object-Guided Dual-Path Decoder.

Compression	Alignment	Overall
MLP compression	None	97.60
Slot compression	None	97.65
Slot compression	OAA	97.50
Slot compression	TSA	98.20
Slot compression	OAA+TSA	**98.85**

(a) Compression and alignment objectives

Decoder	Overall
Object-only pathway	98.35
Detail-only pathway	97.15
Object-Guided Dual-Path Decoder	**98.85**

(b) Object-Guided Dual-Path Decoder

Compression strategy and alignment objectives. As shown in Table 3, slot compression provides a structured representation for action conditioning. TSA preserves predictive dynamics, while OAA provides complementary spatial regularization; combining both objectives performs best.

Object-Guided Dual-Path Decoder. The object-only pathway preserves entity-level structure, while the detail-only pathway lacks sufficient object organization. Combining both pathways performs best, demonstrating the benefit of structured object representations with complementary visual details. Additional qualitative analysis is provided in the Appendix.

5 CONCLUSION

We introduced STaC-WAM, an asymmetric implicit WAM that addresses the overlooked video-to-action conditioning bottleneck in predictive world models. Instead of directly coupling the capacity of the action expert with rich hierarchical world representations, STaC-WAM introduces a Spatio-Temporal Action-Conditioning (STaC) bridge that transforms predictive representations into compact action-conditioning features while preserving manipulation-relevant information.

The proposed STaC bridge combines object-centric compression with the Object-Guided Dual-Path Decoder. Object-centric slots provide a structured bottleneck for removing redundant visual information, while the detail pathway preserves complementary fine-grained visual cues that may be discarded during compression. Object Assignment Alignment and Temporal Similarity Alignment further regularize the compressed representations by preserving entity-level structure and predictive temporal information without introducing additional inference cost.

Experiments demonstrate the effectiveness of this asymmetric design. On LIBERO, STaC-WAM enables a 6-layer Action DiT to outperform the original 30-layer Fast-WAM baseline, improving the average success rate from 97.6% to 98.85% while achieving a 2.10$\times$ inference speedup. These results reveal that predictive visual modeling and action generation do not require symmetric scaling. With an appropriate video-to-action interface, compact action experts can preserve essential predictive information while achieving substantially more efficient inference.

AI USE STATEMENT

Generative AI tools were used to assist with translation, drafting and editing parts of the manuscript, improving readability, brainstorming presentation choices, discovering and summarizing relevant literature, and formatting references. They were also used to assist with data annotation, including generating semantic descriptions used as training targets, and to suggest edits to software code. All AI-assisted outputs were critically reviewed by the authors. References were verified against their original sources, and suggested code changes were manually inspected and tested. The authors take full responsibility for the final content of this work.

REFERENCES

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, Hongyan Zhao, Hanyu Liu, Zhizhong Su, Lei Ma, Hang Su, and Jun Zhu. Motus: A unified latent action world model. In *CVPR*, June 2026.

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, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *CoRL*, 2025a.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *RSS*, June 2025b. doi: 10.15607/RSS.2025.XXI.010.

Jialei Chen, Kai Wang, Kang Chen, Shuaihang Chen, Feng Gao, Wenhao Tang, Zhiyuan Li, Weilin Liu, Zhuyu Yao, Boxun Li, et al. Lawam: Latent world action models for efficient dynamics-aware robot policies. *arXiv preprint arXiv:2606.15768*, 2026a.

Yang Chen, Xiaoguang Ma, and Bin Zhao. Mean-flow based one-step vision-language-action. *arXiv preprint arXiv:2603.01469*, 2026b.

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. In *RSS*, 2024.

Bing Hu, Zaijing Li, Rui Shao, Junda Chen, April Hua Liu, Wei-Shi Zheng, and Liqiang Nie. From Abstraction to Instantiation: Learning Behavioral Representation for Vision-Language-Action Model. In *ICML*, July 2026a. URL https://openreview.net/forum?id=2NRrsz4nUB.

Yucheng Hu, Jianke Zhang, Yuanfei Luo, Yanjiang Guo, Xiaoyu Chen, Xinshu Sun, Kun Feng, Qingzhou Lu, Sheng Chen, Yangang Zhang, Wei Li, and Jianyu Chen. BagelVLA: Enhancing Long-Horizon Manipulation via Interleaved Vision-Language-Action Generation. In *RSS*, 2026b.

Yutong Hu, Jan-Nico Zaech, Nikolay Nikolov, Yuanqi Yao, Sombit Dey, Giuliano Albanese, Renaud Detry, Luc Van Gool, and Danda Pani Paudel. AR-VLA: Autoregressive Action Expert for Vision–Language–Action Models. In *RSS*, 2026c.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *RSS*, June 2025a. doi: 10.15607/RSS.2025.XXI.017.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *CoRL*, 2025b.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos policy: Fine-tuning video models for visuomotor control and planning. In *ICLR*, 2026. URL https://openreview.net/forum?id=wPEIStHxYH.

Hao Li, Qi Lv, Rui Shao, Xiang Deng, Yinchuan Li, Jianye HAO, and Liqiang Nie. Star: Learning diverse robot skill abstractions through rotation-augmented vector quantization. In *ICML*, 2025a.

Jiajun Li, Tiecheng Guo, Yifan Ye, Rongyu Zhang, Xiaowei Chi, Qianpu Sun, Ying Li, Yunfan Lou, Yan Huang, Zhihe Lu, Meng Guo, and Shanghang Zhang. Efficient-WAM: A 1b-parameter world-action model with low-cost future imagination. *arXiv preprint arXiv:2606.10040*, 2026a.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Zhangluyao, Mingrui Yu, Zelin Gao, Nan Xue, Boyu Zhou, Xing Zhu, Mingyu Ding, Yujun Shen, and Yinghao Xu. Causal world modeling for robot control. In *RSS*, 2026b.

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified Video Action Model. In *RSS*, June 2025b. doi: 10.15607/RSS.2025.XXI.074.

Wei Li, Renshan Zhang, Rui Shao, Jie He, and Liqiang Nie. Cogvla: Cognition-aligned vision-language-action model via instruction-driven routing & sparsification. In *NeurIPS*, 2025c.

Zaijing Li, Bing Hu, Rui Shao, Gongwei Chen, Dongmei Jiang, Pengwei Xie, Jianye Hao, and Liqiang Nie. Global Prior Meets Local Consistency: Dual-Memory Augmented Vision-Language-Action Model for Efficient Robotic Manipulation. In *CVPR*, June 2026c.

Ziang Li, Dongzhou Cheng, Yibin Wang, Shiyue Wang, Xiaoyang Xu, Lingxuan Weng, Juan Wang, and Jiaqi Wang. Light-wam: Efficient world action models with state-fusion action decoding. *arXiv preprint arXiv:2606.08242*, 2026d.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. In *NeurIPS*, 2020.

Teli Ma, Jia Zheng, Zifan Wang, Chunli Jiang, Andy Cui, Junwei Liang, and Shuo Yang. DiT4DiT: Jointly modeling video dynamics and actions for generalizable robot control. *arXiv preprint arXiv:2603.10448*, 2026.

Chaojun Ni, Cheng Chen, Xiaofeng Wang, Zheng Zhu, Wenzhao Zheng, Boyuan Wang, Tianrun Chen, Guosheng Zhao, Haoyun Li, Zhehao Dong, Qiang Zhang, Yun Ye, Yang Wang, Guan Huang, and Wenjun Mei. SwiftVLA: Unlocking spatiotemporal dynamics for lightweight vla models at minimal overhead. In *CVPR*, June 2026.

Delin Qu, Haoming Song, Qizhi Chen, Zhaoqing Chen, Xianqiang Gao, Xinyi Ye, Qi Lv, Modi Shi, Guanghui Ren, Cheng Ruan, et al. Eo-1: Interleaved vision-text-action pretraining for general robot control. *arXiv preprint arXiv:2508.21112*, 2025a.

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*, 2025b.

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

Mehdi SM Sajjadi, Daniel Duckworth, Aravindh Mahendran, Sjoerd Van Steenkiste, Filip Pavetic, Mario Lucic, Leonidas J Guibas, Klaus Greff, and Thomas Kipf. Object scene representation transformer. In *NeurIPS*, 2022.

Maximilian Seitzer, Max Horn, Andrii Zadaianchuk, Dominik Zietlow, Tianjun Xiao, Carl-Johann Simon-Gabriel, Tong He, Zheng Zhang, Bernhard Schölkopf, Thomas Brox, and Francesco Locatello. Bridging the gap to real-world object-centric learning. In *ICLR*, 2023. URL `https://openreview.net/forum?id=b9tUk-f_aG`.

Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, Min Cao, Peng Li, Qiuping Deng, Wenjun Mei, Xiaofeng Wang, Xinze Chen, Xinyu Zhou, Yang Wang, Yifan Chang, Yifan Li, Yukun Zhou, Yun Ye, Zhichao Liu, and Zheng Zhu. GigaWorld-Policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026a.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Naveed Malik, Kyungmin Lee, William Liang, Nadun Ranawaka Arachchige, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Danfei Xu, Yilun Du, Ryan Julian, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi Fan, and Joel Jang. World action models are zero-shot policies. In *ICLR Workshop on World Models*, 2026b. URL https://openreview.net/forum?id=cd33uUB609.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-WAM: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

Yang Yue, Yulin Wang, Bingyi Kang, Yizeng Han, Shenzhi Wang, Shiji Song, Jiashi Feng, and Gao Huang. Deer-vla: Dynamic inference of multimodal large language models for efficient robot execution. In *NeurIPS*, 2024.

Andrii Zadaianchuk, Maximilian Seitzer, and Georg Martius. Object-centric learning for real-world videos by predicting temporal feature similarities. In *NeurIPS*, 2023. doi: 10.52202/075280-2689.

Rongyu Zhang, Menghang Dong, Yuan Zhang, Liang Heng, Xiaowei Chi, Gaole Dai, Li Du, Dan Wang, Yuan Du, and Shanghang Zhang. Mole-vla: Dynamic layer-skipping vision language action model via mixture-of-layers for efficient robot manipulation. In *AAAI*, 2026a.

Yuyang Zhang, Wenyao Zhang, Zekun Qi, He Zhang, Haitao Lin, Jingbo Zhang, Yao Mu, Xiaokang Yang, Wenjun Zeng, and Xin Jin. Imagewam: Do world action models really need video generation, or just image editing? *arXiv preprint arXiv:2606.19531*, 2026b.

This supplementary material provides additional evidence for the proposed Spatio-Temporal Action-Conditioning (STaC) bridge. We first analyze the compression and object-path assignments, the slot-mediated spatial prior and detail-path attention, the token-wise routing between the two decoder pathways, and the layer-wise similarity structure of the predictive world representations. We then report extended comparisons on all RoboTwin tasks, provide partial VLABench results, and present representative real-robot execution sequences.

A Qualitative Analysis of STaC

The following visualizations inspect the internal representations produced by STaC using the terminology defined in the main paper. In particular, we visualize the compression assignments $A_g^{\mathrm{cmp},\xi}$, the object-path assignments $A_g^{\mathrm{obj},\xi}$, the slot-mediated spatial prior Π_g^{ξ}, and the token-wise routing weights $w_g^{\mathrm{obj},\xi}$ and $w_g^{\mathrm{det},\xi}$. These visualizations clarify how structural information is compressed, transferred, and supplemented during dense action-conditioning reconstruction. They do not assign fixed semantic meanings to individual slots or attention channels.

For spatial maps without a shared color bar, values are normalized within each panel; their spatial patterns should therefore be compared rather than their absolute magnitudes. Routing and similarity plots retain the numerical scales shown by their corresponding color bars.

A.1 Object Assignment Pseudo-Labels

The Object Assignment Alignment objective does not use manually annotated instance segmentation masks. Instead, we generate object-slot assignment pseudo-labels offline using VideoSAUR with DINOv2 visual features. The exact DINOv2 backbone variant cannot be determined from the retained preprocessing artifacts and is therefore not specified.

VideoSAUR produces three object slots for each video frame. During STaC-WAM training, the pseudo-label corresponding to the first frame of the current observation window is used to supervise the learned token-to-slot assignments. The key and value branches independently produce three slots for each compression group; the three slots are shared across attention heads rather than generated separately by every head. All VideoSAUR pseudo-labels are precomputed and loaded during training, so VideoSAUR introduces no additional inference cost.

A.2 Object-Centric Compression and Reconstruction

Figure 6 compares the compression assignments with the object-path assignments used during reconstruction. The four rows show different manipulation examples with distinct target objects and scene layouts. The compression assignments organize the grouped world features into broad, spatially coherent regions around the robot, candidate objects, and surrounding workspace. The object pathway subsequently retrieves information from these compact slots to reconstruct position-aligned features.

The two assignment maps are not expected to match pixel by pixel. $A_g^{\mathrm{cmp},\xi}$ describes how source tokens are compressed into slots, whereas $A_g^{\mathrm{obj},\xi}$ describes how reconstructed positions attend to those slots. Their broad spatial correspondence indicates that object-centric organization is transferred from compression to reconstruction. The norm of the reconstructed action-conditioning features remains spatially nonuniform and responds strongly around central interaction regions, showing that dense features can be recovered without discarding the compact slot structure.

A.3 Slot-Mediated Spatial Prior and Detail-Path Attention

Figure 7 visualizes the slot-mediated spatial prior and detail-path attention at four sampled stages from each of four task families. As defined in the main paper, Π_g^{ξ} is constructed from the shared slot correspondence between the compression assignments and the object-path assignments. It transfers object-consistent spatial correspondence from the original world features to the reconstructed positions.

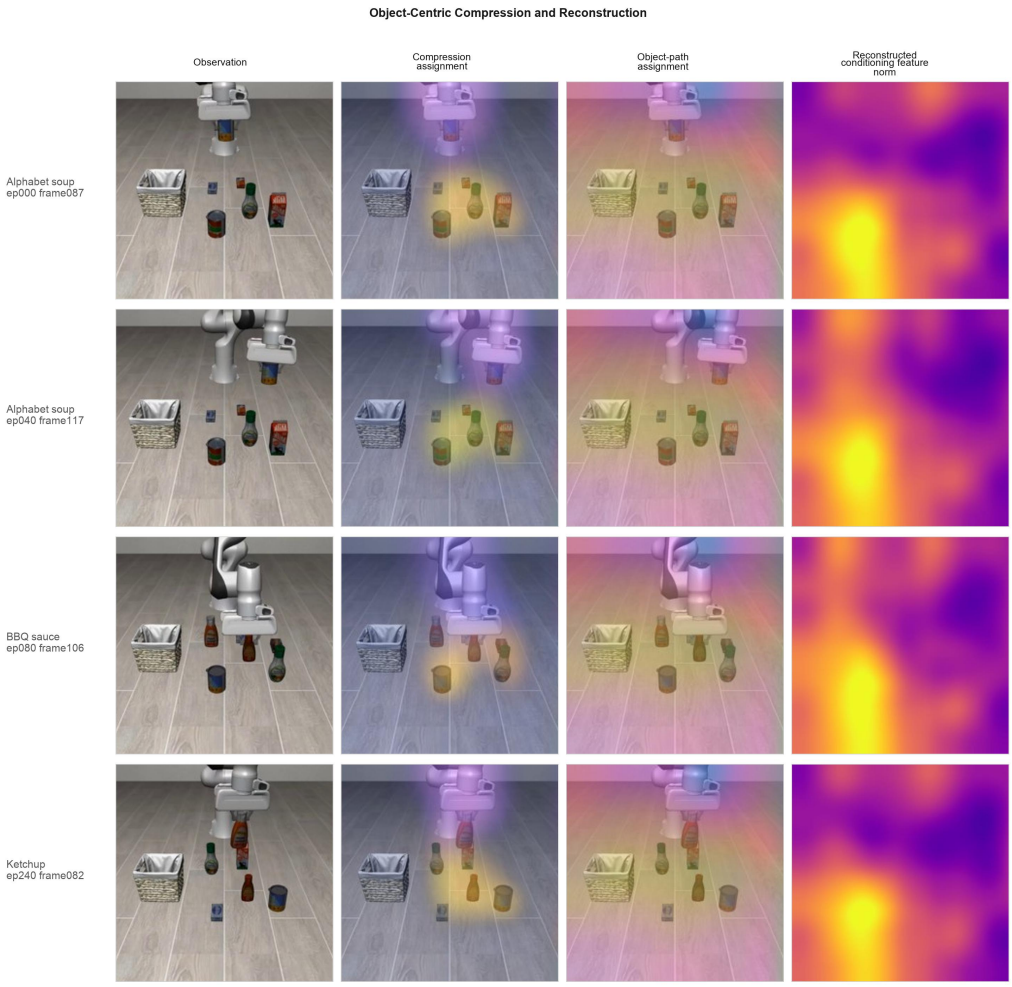


Figure 6: Object-centric compression and reconstruction. Each row shows an observation, compression assignment $A_g^{\mathrm{cmp},\xi}$, object-path assignment $A_g^{\mathrm{obj},\xi}$, and the norm of the reconstructed action-conditioning features. The assignments exhibit related spatial organization, indicating that the object pathway reuses the compact slots during dense feature reconstruction. Colors represent relative slot contributions within each example and do not imply fixed slot semantics across different episodes.

Across the displayed stages, the slot-mediated spatial prior provides a broad spatial scaffold that changes with the robot configuration and object locations while continuing to cover the interaction region. The detail-path attention is more selective and shifts between localized regions as the manipulation progresses. This distinction is consistent with the intended roles of the two components: the slot-mediated spatial prior preserves object-level organization, while the detail pathway retrieves complementary content from the aggregated source features. The detail pathway therefore does not simply model non-object or background regions; it supplements localized visual information under object-guided spatial constraints.

A.4 Token-Wise Routing between the Two Decoder Pathways

Figure 8 analyzes the token-wise routing weights over the six conditioning groups and 98 reconstructed positions. The upper panels show the contributions assigned to the object pathway and detail pathway for the reconstructed key and value caches. The token-deviation maps further show that the routing decision is not a single constant for each conditioning group: tokens within the same group can increase or decrease their object-path contribution according to local content.

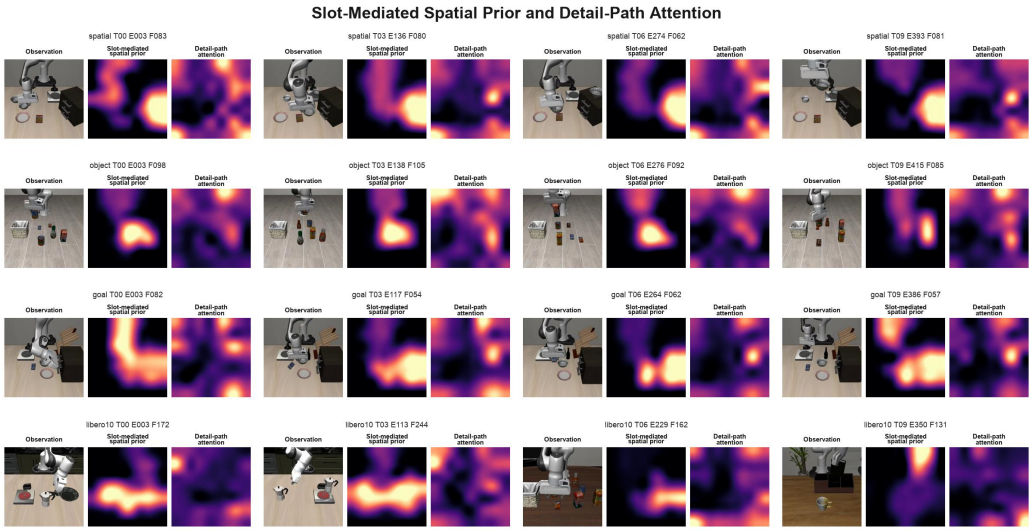


Figure 7: Slot-mediated spatial prior and detail-path attention across rollout stages. Each task row contains four sampled stages, and every stage shows the observation, slot-mediated spatial prior Π_g^{ξ}, and detail-path attention. The prior provides a broad object-guided spatial scaffold, whereas the detail-path attention changes more selectively with the current scene and interaction state.

The aggregate statistics reveal a clear asymmetry between keys and values. For the reconstructed key cache, the mean detail-path contribution ranges from approximately 0.56 to 0.82, while the object-path contribution ranges from 0.18 to 0.44. For the reconstructed value cache, the trend is reversed: object-path contributions remain between approximately 0.67 and 0.73, with detail-path contributions between 0.27 and 0.33. Therefore, the token-wise router does not collapse both cache branches to a shared fixed mixture. It retains more complementary source detail in the keys while relying more strongly on object-centric slot features in the values, with additional token-level adaptation inside each conditioning group.

A.5 Layer-Wise Token Similarity of Predictive Representations

Figure 9 visualizes token-level cosine-similarity matrices for all 30 predictive world layers. Repeated diagonal and banded structures appear throughout the world expert, indicating that neighboring and corresponding tokens retain organized relationships across depth. At the same time, the detailed pattern changes between layers: some matrices exhibit smooth local bands, whereas others contain stronger block-like, periodic, or long-range structures.

The visualization highlights both redundancy and hierarchy in the original world representations. Adjacent layers share substantial structural regularity, supporting their aggregation into compact conditioning groups, but the representations are not identical across depth. This motivates a learned video-to-action transformation that removes redundant information while preserving layer-dependent token relationships relevant to action generation.

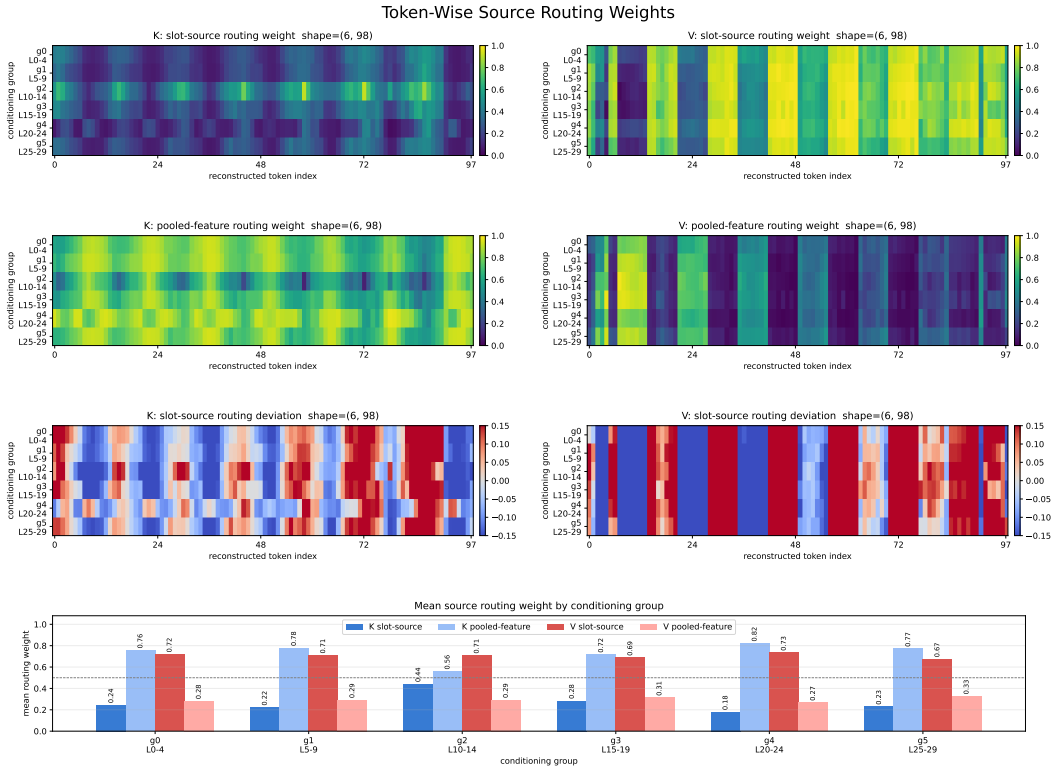


Figure 8: Token-wise routing between the object and detail pathways for the reconstructed key and value caches. The first two rows show the routing weights assigned to the object pathway and detail pathway for each conditioning group and reconstructed position. The third row shows token-level deviations from the corresponding group mean, and the bottom panel summarizes the mean pathway contributions. Keys retain a larger detail-path contribution, whereas values rely more strongly on the object pathway, demonstrating cache-specific and token-adaptive fusion.

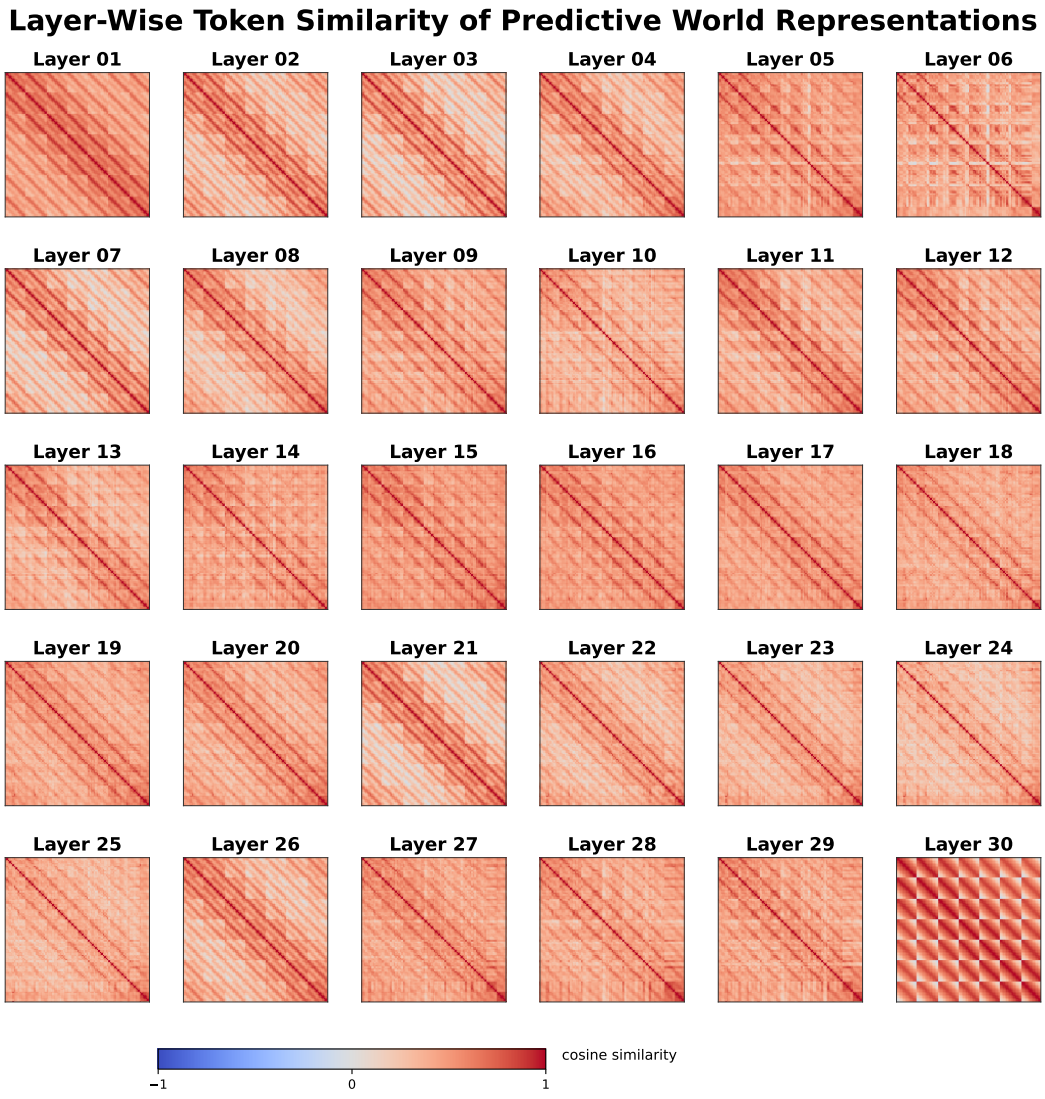


Figure 9: Layer-wise token similarity of the predictive world representations. Each panel shows the token-level cosine-similarity matrix of one world layer. Diagonal, banded, and block-like structures persist across depth, while their detailed patterns vary between layers. The visualization reveals substantial structural regularity together with layer-dependent organization in the original hierarchical representations.

Table 4: Performance comparison on RoboTwin. We report per-task success rates (%) under both clean and randomized settings. Best results in each row are highlighted in bold.

Task	FastWAM (Yuan et al., 2026)		FastWAM-Joint (Yuan et al., 2026)		FastWAM-IDM (Yuan et al., 2026)		LingBot-VA (Li et al., 2026b)		$\pi_{0.5}$ (Black et al., 2025a)		Motus (Bi et al., 2026)		STaCWAM	
	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.
Adjust Bottle	**100**	**100**	98	99	94	99	90	94	**100**	99	89	93	**100.0**	99.0
Beat Block Hammer	99	97	**100**	**98**	98	**98**	96	**98**	96	93	95	88	**100.0**	**98.0**
Blocks Ranking RGB	**100**	**100**	**100**	**100**	**100**	99	99	98	92	85	99	97	99.0	99.0
Blocks Ranking Size	**94**	98	83	91	79	90	**94**	96	49	26	75	63	**94.0**	**99.0**
Click Alarmclock	**100**	**100**	**100**	**100**	98	**100**	99	**100**	98	89	**100**	**100**	**100.0**	99.0
Click Bell	**100**	**100**	**100**	98	**100**	96	**100**	**100**	99	66	**100**	**100**	**100.0**	99.0
Dump Bin Bigbin	**97**	96	95	95	93	**98**	89	96	92	97	95	91	96.0	97.0
Grab Roller	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100.0**	99.0
Handover Block	95	81	93	91	97	**94**	**99**	78	66	57	86	73	96.0	82.0
Handover Mic	99	**100**	**100**	**100**	98	99	94	96	98	97	78	63	**100.0**	**100.0**
Hanging Mug	58	62	**71**	56	66	62	40	28	18	17	38	38	57.0	**63.0**
Lift Pot	**100**	**100**	**100**	**100**	**100**	**100**	**100**	99	96	85	96	99	**100.0**	**100.0**
Move Can Pot	90	88	**97**	99	**97**	**100**	94	97	51	55	34	74	91.0	89.0
Move Pillbottle Pad	**100**	99	99	**100**	98	**100**	99	99	84	61	93	96	**100.0**	**100.0**
Move Playingcard Away	**100**	**100**	**100**	**100**	99	**100**	**100**	99	96	84	**100**	96	99.0	**100.0**
Move Stapler Pad	77	64	85	81	89	**85**	**91**	79	56	42	83	**85**	77.0	65.0
Open Laptop	98	**100**	89	92	92	92	92	94	90	96	95	91	**99.0**	**100.0**
Open Microwave	62	45	3	14	54	53	82	86	34	77	**95**	**91**	62.0	46.0
Pick Diverse Bottles	80	85	86	87	87	89	89	82	81	71	**90**	**91**	79.0	86.0
Pick Dual Bottles	**100**	96	98	**99**	**100**	98	**100**	**99**	93	63	96	90	**100.0**	97.0
Place A2B Left	95	93	96	**96**	**97**	**96**	**97**	93	87	82	88	79	96.0	94.0
Place A2B Right	93	99	95	95	94	98	**97**	95	87	84	91	87	93.0	**100.0**
Place Bread Basket	91	93	89	94	91	**97**	**97**	95	77	64	91	94	93.0	93.0
Place Bread Skillet	90	93	90	93	90	**95**	**95**	90	85	66	86	83	90.0	93.0
Place Burger Fries	96	99	**100**	**100**	97	99	97	95	94	87	98	98	98.5	98.5
Place Can Basket	71	69	50	23	37	28	**81**	**84**	62	62	**81**	76	71.0	70.0
Place Cans Plasticbox	99	96	98	98	98	96	**100**	**99**	94	84	98	94	98.0	96.0
Place Container Plate	96	**100**	99	98	**100**	96	99	97	99	95	98	99	99.0	99.0
Place Dual Shoes	94	88	93	**89**	85	87	94	**89**	75	75	93	87	**95.0**	**89.0**
Place Empty Cup	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	99	99	98	**100.0**	**100.0**
Place Fan	96	**96**	**99**	**96**	97	95	**99**	93	87	85	91	87	95.0	**96.0**
Place Mouse Pad	83	89	96	91	**97**	93	93	**96**	60	39	66	68	83.0	90.0
Place Object Basket	89	88	86	81	87	82	**91**	88	80	76	81	87	90.0	**90.0**
Place Object Scale	90	97	96	**99**	**99**	**99**	96	95	86	80	88	85	90.0	97.0
Place Object Stand	90	94	92	98	96	**100**	**99**	96	91	85	98	97	94.0	94.0
Place Phone Stand	97	99	**100**	**100**	99	99	97	97	81	81	87	86	97.0	**100.0**
Place Shoe	96	**99**	95	97	95	98	98	98	92	93	**99**	97	97.0	**99.0**
Press Stapler	90	97	52	50	50	57	85	82	87	83	**93**	**98**	90.0	**98.0**
Put Bottles Dustbin	95	90	93	**95**	**97**	92	87	91	84	79	81	79	94.0	90.0
Put Object Cabinet	94	89	**95**	**90**	93	**90**	85	87	80	79	88	71	94.0	**90.0**
Rotate QRcode	93	89	91	**92**	91	86	**96**	91	89	87	89	73	94.0	89.0
Scan Object	89	92	92	92	93	90	**96**	91	72	65	67	66	89.0	**93.0**
Shake Bottle	**100**	**100**	**100**	**100**	**100**	**100**	**100**	97	99	97	**100**	97	99.0	**100.0**
Shake Bottle Horizontally	**100**	**100**	**100**	**100**	**100**	**100**	**100**	99	99	99	**100**	98	**100.0**	**100.0**
Stack Blocks Three	95	97	98	97	**99**	95	**99**	**98**	91	76	91	95	96.0	97.0
Stack Blocks Two	**100**	**100**	**100**	**100**	**100**	**100**	**100**	98	97	**100**	**100**	98	**100.0**	**100.0**
Stack Bowls Three	80	81	84	86	85	83	**86**	83	77	71	79	**87**	79.0	82.0
Stack Bowls Two	92	**98**	97	95	94	96	94	**98**	95	96	**98**	**98**	97.0	97.0
Stamp Seal	90	94	96	**99**	**99**	94	96	97	79	55	93	92	91.0	94.0
Turn Switch	61	59	73	72	59	74	44	45	62	54	**84**	**78**	61.0	60.0
Average	91.88	91.78	90.84	90.32	91.16	91.34	**92.90**	91.50	82.74	76.76	88.66	87.02	92.25	**92.11**

B Additional Results on VLABench

B.1 Tasks and Evaluation Protocol

We additionally conduct a partial evaluation on eight VLABench tasks. The subset mainly covers language-conditioned object selection and object interaction, including selecting books, paintings, mahjong tiles, fruits, drinks, and chemistry tubes, as well as adding condiment and inserting a flower. STaC-WAM and STaC-WAM are evaluated using the same task instances, observations, instructions, and scoring protocol.

The subset was selected to provide a compact cross-section of primitive language-conditioned manipulation rather than to reproduce the full scope of VLABench. It primarily evaluates object selec-

tion across visually and semantically distinct categories, including books, paintings, mahjong tiles, fruits, drinks, and chemistry tubes. We additionally include two goal-directed interaction tasks, adding condiment and inserting a flower, so that the evaluation is not limited to recognition-driven selection. The selected tasks also span a broad empirical difficulty range under STaC-WAM, with success rates ranging from 18% to 80%, covering both frequently completed tasks and challenging cases with substantial partial progress but limited final success. Together, these tasks examine visual grounding, instruction-conditioned target identification, and precise object interaction across heterogeneous objects and completion levels. Because the subset consists of primitive tasks and does not cover composite long-horizon scenarios, it should not be interpreted as representative of all reasoning capabilities evaluated by VLABench.

We report success rate (SR), intention score (IS), and progress score (PS). Since this evaluation covers only eight tasks, the resulting averages are reported as selected-task averages and should not be interpreted as complete VLABench scores.

B.2 Selected-Task Results

Table 5: Partial evaluation on eight selected VLABench tasks. We report success rate (SR), intention score (IS), and progress score (PS). Bold values indicate the better result between STaC-WAM and STaC-WAM for each metric. The final row reports the average only over the displayed tasks.

Task	SR (%)		IS (%)		PS (%)	
	FastWAM	STaC-WAM	FastWAM	STaC-WAM	FastWAM	STaC-WAM
`select_chemistry_tube`	64.0	**76.0**	0.0	0.0	77.0	**86.0**
`select_book`	70.0	**76.0**	**100.0**	98.0	84.0	**87.0**
`select_painting`	18.0	**22.0**	100.0	100.0	18.0	**22.0**
`select_mahjong`	76.0	**80.0**	100.0	100.0	**87.0**	84.0
`add_condiment`	76.0	**80.0**	100.0	100.0	**88.7**	88.0
`select_fruit`	26.0	26.0	100.0	100.0	**62.0**	60.0
`insert_flower`	**42.0**	38.0	100.0	100.0	**71.0**	69.0
`select_drink`	**80.0**	72.0	**98.0**	94.0	**87.0**	78.0
Selected-task Average	56.50	**58.75**	**87.25**	86.50	**71.84**	71.75

Across the eight selected tasks, STaC-WAM improves average success rate from 56.50% to 58.75%. It obtains higher SR on five tasks, ties on `select_fruit`, and performs worse on two tasks. The largest gain is observed on `select_chemistry_tube`, where SR increases from 64.0% to 76.0% and PS increases from 77.0% to 86.0%. Improvements are also observed on `select_book`, `select_painting`, `select_mahjong`, and `add_condiment`.

The selected-task averages for IS and PS remain close to STaC-WAM, decreasing by 0.75 and 0.09 percentage points, respectively. This indicates that the compressed interface broadly preserves instruction-level and partial-progress behavior while modestly improving final task completion on this subset. However, the decline on `insert_flower` and especially `select_drink` shows that the gains are not uniform across tasks. These results should therefore be interpreted as preliminary cross-benchmark evidence rather than a complete VLABench evaluation.

C Real-Robot Rollout Snapshots

The main paper reports the quantitative real-robot evaluation. Here, we provide representative rollout snapshots to show the execution process for each task. The snapshots serve as qualitative examples rather than additional task-level statistics. Within each row of Figure 10, frames are ordered chronologically from left to right.

C.1 Task Descriptions

T1: Object Sorting. The robot identifies objects placed in the wrong target region and returns them to the corresponding left or right tray.

T2: Retrieval under an Inverted Cup. The robot locates and retrieves a block hidden beneath an inverted cup.

T3: Retrieval under Cloth. The robot retrieves a fruit that is partially occluded beneath a cloth.

T4: Color-Conditioned Fruit Selection. The robot observes the light color and selects the fruit associated with the corresponding visual condition.

T5: Conveyor-Belt Grasping. The robot detects and grasps a moving object from a conveyor belt, requiring visual tracking and timely action generation.

C.2 Execution Sequences

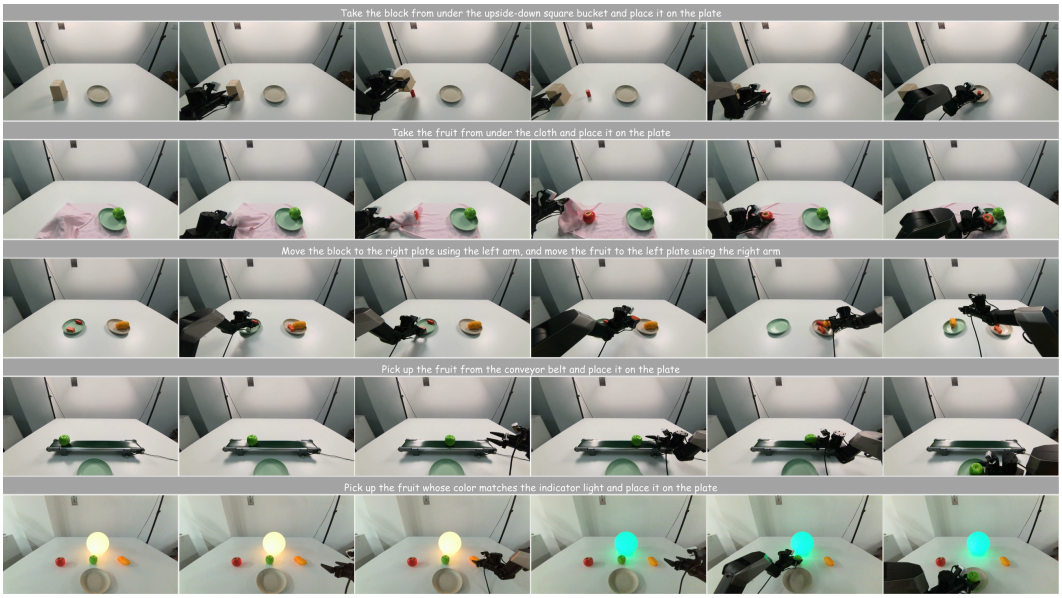


Figure 10: Representative real-robot rollout snapshots. Each row corresponds to one task, and frames within each row are ordered chronologically from left to right. T2 retrieves a block beneath an inverted cup; T3 retrieves a fruit under cloth occlusion; T1 performs object sorting; T5 grasps a moving object from a conveyor belt; and T4 performs color-conditioned fruit selection. The figure complements the quantitative results in the main paper by illustrating the progression from initial observation to task completion.

The sequences cover complementary real-world challenges. T2 and T3 require the policy to act under partial visual occlusion, while T4 requires a visual attribute to be connected to the appropriate object-level action. T5 further requires closed-loop adjustment to a moving target. Together, the examples show that the compact predictive conditioning interface can support object reasoning, occlusion handling, and dynamic interaction in real scenes.