COPPER-POLICY: FOCUS ON THE REPRESENTATION FOR ROBUST ROBOT MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

World Action Models (WAMs) acquire behavioral priors by modeling future scene evolution, but predicting detailed futures in pixel or latent space incurs substantial cost. Recent evidence that co-training gains persist without test-time generation raises a question: what must a WAM learn to improve control? We introduce ***Copper-Policy***, which learns a compact World representation with the policy rather than relying on a predefined target space. Through temporal joint-embedding prediction, it predicts future observation embeddings conditioned on task intention without reconstructing pixels. This prediction and action decoding shape the representation jointly, while the policy retains access to current-frame spatial detail for execution. Representation analyses show that the learned features better separate task-driven change from perturbations and provide complementary information for control. Compact prediction targets reduce training tokens per sample, **enabling a 2B-parameter model trained in 9.67 hours on 8× RTX 5090 GPUs and 6× faster than Fast-WAM on matched A100 GPUs**. *Copper-Policy* outperforms every compared method without embodied pretraining on RoboTwin and several embodied-pretrained VLAs on LIBERO-Plus (80.85%). On three challenging real-robot tasks, it performs comparably to $\pi_{0.5}$ and attains a higher average score. Together, these results show that Copper-Policy combines strong control performance with efficient training.

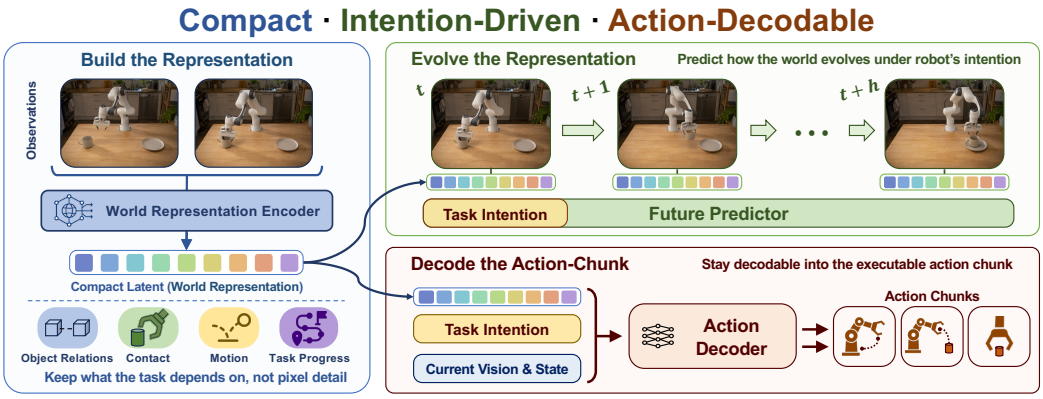


Figure 1: **Representation learning illustration.** Joint-embedding prediction and action decoding jointly shape a compact World representation, enabling strong performance with low training cost.

1 INTRODUCTION

Robotic research has long pursued autonomous systems equipped with human-like cognition to enable daily physical interaction. This pursuit has been accelerated by recent advances in the concept of ***world modeling***, which offers a shared substrate for building decision-oriented cognition that can both *comprehend* and *predict what will happen*, specialized for downstream embodiment control. One prominent backbone is the Vision-Language-Action (VLA) model, which capitalizes on the semantic capabilities of pretrained Vision-Language Models (VLMs) to establish a direct mapping

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.

from visual observations and language instructions to executable robot actions (Kim et al., 2025; Physical Intelligence et al., 2025; Zheng et al., 2026; Yang et al., 2026c). In contrast, the emerging line, World Action Models (WAMs), go further by modeling future scene evolution through visual dynamics (Yuan et al., 2026; Li et al., 2026), explicitly linking *what the world should become* to *how to achieve it*. Yet despite their effectiveness, generative WAMs couple action with explicit video (or latent video) generation, leading to substantial training burden and inference overhead.

This limitation necessitates a reassessment: ***what aspect of dynamics learning in WAM actually drives better control?*** Fast-WAM (Yuan et al., 2026) demonstrates that video co-training remains beneficial even without test-time future generation, while more recent studies indicate that the learned embeddings that encode informative future representations could actively shape policy decisions (Zheng et al., 2025; Su et al., 2026; Luo et al., 2026). The value of WAMs thus appears to lie less in the futures they generate than in the representations their training shapes, motivating a latent-prediction paradigm that predicts future states directly in representation space. Existing designs largely fall into two categories. One line uses a pretrained Joint-Embedding Predictive Architecture (JEPA) space either as additional policy input or as fixed prediction targets (Sun et al., 2026; Lin et al., 2026). The other constructs a predictive latent under fixed constraints: LiLa-WAM (Yang et al., 2026a) compresses current features with learned queries but decodes its predicted future into frozen DINOv3 features, while StageWAM (Liu et al., 2026a) freezes V-JEPA2 on both sides and then freezes its predictor before policy learning. In both cases, the predictive target space does not co-evolve with the policy that ultimately consumes it.

To instantiate this perspective, we introduce ***Copper-Policy***, which treats predictive representation learning as part of the policy itself. Rather than requiring the model to reconstruct detailed futures, Copper-Policy distills task-relevant scene evolution into a compact latent representation and uses it to guide action generation. The representation is therefore encouraged to capture *what is changing and where the task is heading*, while execution-specific spatial detail can remain available to the Action Expert. In this way, joint-embedding prediction serves as a mechanism for organizing representations for control, rather than as an additional generative objective.

Contributions.

- **Policy-learned predictive target space.** We formulate latent world modeling such that the predictive target space is learned jointly with the policy, rather than prescribed by a pretrained encoder or a staged representation. This lets predictive learning and action supervision jointly organize the representation used for control.
- **Asymmetric prediction–control interface.** We design the Future Expert to operate only on the compact predictive representation, while the Action Expert additionally accesses current-frame features. This allows the predictive target space to focus on task-relevant evolution without being forced to retain all execution-level spatial detail.
- **Empirical and representation-level evidence.** Across simulation and real-robot evaluations, Copper-Policy combines strong control performance with efficient training. Ablations and representation diagnostics link robustness to common closed-loop shifts to the learned World representation and show the complementary role of current-frame features.

2 Related Work

A broad line of work augments robot policy learning with signals derived from future observations or prediction, spanning world action models, future-conditioned policies, and joint-embedding prediction. A key distinction is the target space, which determines what information the representation retains for policy. Our comparison concerns representations that the policy reads for action decoding. In Copper-Policy, the predictive target space is the same compact World representation read by the Action Expert.

Fixed target spaces preserve information beyond what control requires. Fast-WAM (Yuan et al., 2026), LingBot-VA (Li et al., 2026), and Cosmos-Policy (Kim et al., 2026) predict future visual targets in pretrained video-generation latents, whereas earlier video-prediction policies generate future images or videos directly (Du et al., 2023; Ko et al., 2024; Cheang et al., 2024). Both substrates preserve information needed for visual synthesis and may retain appearance not relevant

to action prediction. Other WAM targets include future RGB–D videos (Guo et al., 2026), evolving geometric relations (Zhang et al., 2026a), and RGB–depth–flow futures (Zhao et al., 2026). Outside WAMs, the same pattern appears with targets that are not futures at all (Li et al., 2025).

Control-oriented targets compress future information, but are typically pretrained or staged. A second line uses future observations to learn control-oriented policy states. FLARE (Zheng et al., 2025) carries future tokens through action denoising and aligns them with the future-observation embedding of an action-aware vision-language encoder, pretrained in an earlier stage and kept as a stop-gradient EMA target encoder during policy training. WoG (Su et al., 2026) first learns a compact future condition through action prediction, then freezes it while training the current-context pathway to predict that condition. Being-H0.7 (Luo et al., 2026) instead jointly aligns current latent queries with a future-informed posterior inside the action model. Together, these methods show that privileged future information can shape compact, action-relevant policy states without test-time rollout. Our predictive target space is learned jointly with the policy. Current and future representations inhabit the same compact space, which action decoding also uses and shapes alongside prediction. Robot-relevant visual representations can also be pretrained on human and other Internet videos through time-contrastive (Nair et al., 2023), value-based (Ma et al., 2023), masked (Radosavovic et al., 2023), or language-driven (Karamcheti et al., 2023) objectives and then frozen. A separate line learns latent *actions* from unlabeled video (Ye et al., 2025; Bruce et al., 2024), encoding transitions rather than a compact observation representation.

JEPA predicts in representation space, but prior robot policies do not jointly shape that space through control. Joint-embedding predictive architectures (JEPAs) predict targets in representation space rather than reconstructing pixels. I-JEPA (Assran et al., 2023) and V-JEPA (Bardes et al., 2024) use momentum-updated target encoders to define masked representation targets in images and videos, learning visual representations without an action objective and with deterministic predictors. VLA-JEPA (Sun et al., 2026) first pretrains a student on current observations to predict latent representations produced by a target encoder from future frames, then fine-tunes an action head. Copper-Policy instead learns the predictive target space jointly with action decoding and predicts in that same space by flow matching. Latent world models (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2025) put related predictive structures to a different use. DINO-WM (Zhou et al., 2025) and LeWM (Maes et al., 2026) roll out frozen patch features and a learned latent, conditioned on candidate actions so that a planner can search over them at test time.

Copper-Policy builds on Fast-WAM's two-stream design and its finding that future prediction need not be used at deployment. The key difference is that the predictive target space is learned jointly with action decoding, so prediction shapes the same compact representation used for control.

3 METHOD

3.1 OVERVIEW

Figure 2 gives the architecture. At time t the policy observes o_t, a set of camera views, together with robot state q_t and task instruction s, and produces an action chunk of horizon H_a,

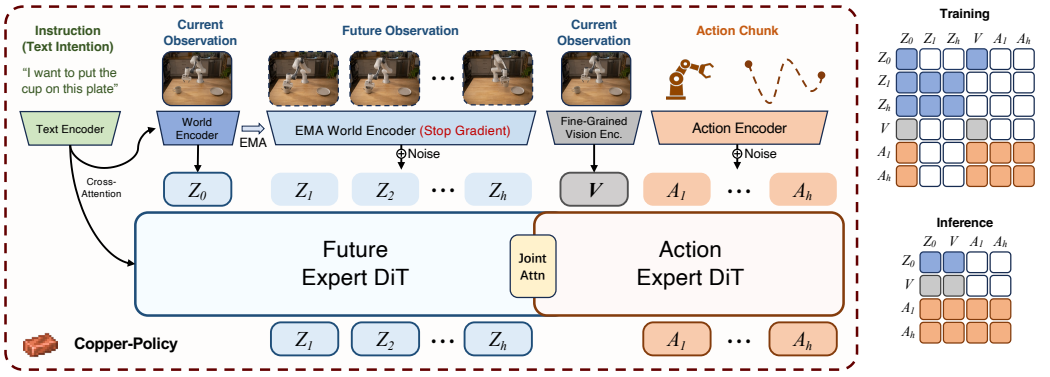
$$a^+ := a_{t:t+H_a-1} \sim p(a^+ \mid o_t, q_t, s). \tag{1}$$

WAMs augment this with prediction targets over a future horizon H_f. Copper-Policy uses temporal joint-embedding prediction to learn a compact space conditioned on task intention and constrained by action decoding, implemented by an online–EMA compactor pair and two expert streams coupled through a structured attention mask. No future is predicted at test time.

3.2 SHARED REPRESENTATION WITH ONLINE AND EMA COMPACTORS

Let f map an observation into a source space, either the features of a frozen pretrained encoder or the pixels themselves, and let $e_s = E_{\text{text}}(s)$ be a task-intention embedding from a text encoder, shared across the Future and Action Experts. The online compactor C_ϕ reads that source space under cross-attention to e_s and writes a compact target space:

$$z_t = C_\phi\big(f(o_t), e_s\big) \in \mathbb{R}^{K\times d}, \tag{2}$$



(a) The Implementation of Copper-Policy (b) The attention masks

Figure 2: **Copper-Policy overview.** An online compactor produces the World representation z_t (blue), while a spatial encoder and grid sampler produce current-frame tokens v_t (gray). An EMA compactor supplies stop-gradient targets $\tilde{z}_{t+1:t+H_f}$. The Future Expert predicts these targets, and the Action Expert decodes the action chunk (orange) with access to the robot state q_t.

that is, K compact tokens of width d. The input layout and the compactor's spatial mask determine K, yielding 28 tokens for LIBERO's multi-view layout and 36 for the T-shaped RoboTwin and real-robot layout. Band-limited queries keep a coarse correspondence between compact tokens and image regions. The formulation leaves the source-to-compact mapping flexible. Here f is a frozen V-JEPA-family encoder (Bardes et al., 2024), while C_ϕ is trainable. Together, f and C_ϕ form the World Encoder of Fig. 2.

We regard o_t and $o_{t+\tau}$ from the same instructed trajectory as current and future views of one scene. Because the compactor reads e_s directly, the compact space can retain what the instructed task depends on while discarding the rest. We call z_t the *World representation*, a compact, task-conditioned behavioral prior assigned to the current view in a shared temporal embedding space. It serves as the *World input* to both expert streams in place of dense visual tokens.

Targets come from an EMA copy $C_{\phi'}$ of the same compactor, updated after every optimizer step with decay m, under stop-gradient:

$$\tilde{z}_{t+\tau} = \mathrm{sg}\big(C_{\phi'}\big(f(o_{t+\tau}), e_s\big)\big)\,, \qquad \tau = 1, \ldots, H_f. \tag{3}$$

Since a learned target space admits collapse, we follow BYOL (Grill et al., 2020) and subsequent joint-embedding practice (Assran et al., 2023). Gradients are stopped through the target branch, and $C_{\phi'}$ tracks C_ϕ by EMA, so the target space co-evolves with the online representation. The frozen f is shared rather than copied. The future horizon matches the action horizon, $H_f = H_a$, and we construct N_f target nodes from this horizon, as detailed in Appendix A.1. We write $\tilde{z}$ for the resulting target sequence and $z^{(u)}$ for its noised counterpart below.

3.3 Temporal Joint-Embedding Prediction

The Future Expert implements temporal joint-embedding prediction by predicting the compact future-view targets $\tilde{z}$ from z_t via conditional flow matching (Lipman et al., 2023), a formulation also used by diffusion and flow policies (Chi et al., 2023; Black et al., 2024). The prediction is conditioned on task intention e_s. We parameterize the flow from the clean target at $u = 0$ to pure noise at $u = 1$. For noise $\epsilon \sim \mathcal{N}(0, I)$, we interpolate

$$z^{(u)} = (1 - u)\,\tilde{z} + u\,\epsilon, \tag{4}$$

and train a velocity field $\pi_\theta(z^{(u)}, u \mid z_t, e_s)$, the Future Expert itself, to match $\epsilon - \tilde{z}$, the derivative of the interpolation path with respect to u. Rather than sampling u uniformly we use a shift-based schedule $u = \varphi(\rho)$, $\rho \sim \mathcal{U}[0, 1]$ (Appendix A.1). The flow-matching Future Expert is one of two streams in a joint self-attention transformer based on the mixture-of-transformers pattern (Liang et al., 2025; Yuan et al., 2026). A structured mask lets it attend only to z_t and other future tokens, not directly to the current-frame tokens v_t or the robot state introduced next.

3.4 Action Expert and Current-Frame Vision

A separate pathway supplements z_t with current-frame spatial detail. A learnable spatial encoder g_ψ feeds a differentiable grid sampler S_ξ (Feng et al., 2026), a visual bottleneck that compresses a feature map into a few tokens, and together they produce $v_t = S_\xi(g_\psi(o_t))$. The Action Expert, the second stream of the joint self-attention transformer, additionally receives the robot state q_t as a single token projected alongside e_s, and decodes the action chunk with the same flow-matching formulation. For $a^{(u)} = (1-u)\,a^+ + u\,\epsilon_a$ it predicts the velocity

$$\hat{v}^a = \pi^a_\theta\Big(a^{(u)}, u \mid z_t, v_t, q_t, e_s\Big), \tag{5}$$

trained to match $\epsilon_a - a^+$ and integrated at test time to give $\hat{a}^+$. Under the same mask, action tokens attend to z_t, to v_t, and to other action tokens, but never to the predicted future tokens. Thus, temporal joint-embedding prediction affects control only through the representations it shapes during training. Because the compactor is shared, the action loss directly constrains the space itself, pressing the World representation to carry what the action decoder reads.

3.5 Asymmetric Interface

This asymmetric interface keeps the predictive target compact while giving control direct access to current-frame detail. The structured mask of Sections 3.3 and 3.4 gives the two objectives unequal readouts. Future queries attend to z_t and other future tokens, whereas action queries attend to z_t, the current-frame tokens v_t, and the state q_t. Within the current-step prefix, z_t and v_t attend to each other, so detail reaches future queries only after passing through the K compact tokens, while action decoding reads v_t directly. Consequently, joint-embedding prediction explains the future from z_t (Grill et al., 2020; Assran et al., 2023), and its targets contain only compactor outputs. Action decoding can therefore use instantaneous detail without requiring z_t to duplicate it, while the action loss still shapes z_t toward information useful for control. Optimization mirrors this asymmetry. The compactor and Future Expert use a $2\times$ learning-rate multiplier relative to the remaining policy parameters, giving the predictive pathway a faster adaptation timescale.

We test two predictions in Section 4.6. Joint-embedding prediction should reduce the overlap between perturbation-induced and task-driven temporal variation in z_t. The detail needed for precise action should live in v_t rather than in z_t, even though the action loss also presses on z_t.

3.6 Compact Token Interface

Each future target step adds a block of target tokens, so the target space sets the sequence length the experts attend over at every training step. In the real-robot setting Copper-Policy processes 244 tokens per sample against 392 for Fast-WAM, and the Future Expert attends over 180 of them against 360 (Appendix A.5). Self-attention is quadratic in that length, so the size of the target space bounds training throughput as the prediction horizon grows.

3.7 Joint Objective and Inference

The temporal joint-embedding loss is

$$\mathcal{L}_{\text{pred}} = \mathbb{E}_{\rho\sim\mathcal{U}[0,1],\ \epsilon\sim\mathcal{N}(0,I)} \left\| \pi_\theta\Big(z^{(u)},\, u \mid z_t, e_s\Big) - (\epsilon - \tilde{z}) \right\|_2^2, \qquad u = \varphi(\rho), \tag{6}$$

and the action loss regresses the same kind of velocity, $\mathcal{L}_{\text{act}} = \mathbb{E}\|\pi^a_\theta(a^{(u)}, u \mid z_t, v_t, q_t, e_s) - (\epsilon_a - a^+)\|_2^2$. We optimize $\mathcal{L} = \mathcal{L}_{\text{act}} + \lambda_{\text{pred}}\mathcal{L}_{\text{pred}}$ end-to-end, with λ_{pred} weighting the predictive term. The EMA compactor $C_{\phi'}$ never receives gradients.

Inference. At test time, the model receives only the current observation o_t, the robot state q_t, and the instruction s. The online compactor and grid sampler produce z_t and v_t. The Future Expert stream runs once as a context encoder to populate its per-layer keys and values. The Action Expert then decodes $\hat{a}^+$ over 10 flow steps, attending jointly over that cache and its own tokens. No future tokens are instantiated.

Table 1: Experimental setup. (a) Per-benchmark data and rollout settings; the action space is either delta end-effector (EEF) pose or absolute joint positions (qpos). (b) *w/o World Input & Future Expert* gives the Action Expert only current-frame features and language.

(a) Per-benchmark settings.

	LIBERO	RoboTwin	Real
Camera views	2	3	3
Action space	delta EEF	abs. qpos	abs. qpos
Demos / task	50	50	300
World tokens K	28	36	36
Rollouts / task	50	100	50

(b) Ablation configurations.

Configuration	z_t	JE Pred.	v_t
Copper-Policy	✓	✓	✓
– w/o Joint-Embedding Prediction	✓	×	✓
– w/o Current-Frame Visual Features	✓	✓	×
– w/o World Input & Future Expert	×	×	✓
– w/o JE Pred. & Visual Features	✓	×	×

4 EXPERIMENTS

We evaluate Copper-Policy on LIBERO and LIBERO-Plus, RoboTwin, and three real-robot manipulation tasks, using the same ablation configurations throughout (Table 1b).

4.1 IMPLEMENTATION DETAILS

Copper-Policy uses no embodied pretraining components. A frozen 300M V-JEPA encoder (Bardes et al., 2024) feeds a trainable 25M compactor that produces the World representation. In parallel, a 300M spatial encoder initialized from DINOv2-L with registers (Oquab et al., 2024; Darcet et al., 2024) is trained with the policy and supplies the grid sampler (Feng et al., 2026) that produces v_t. As in Fast-WAM (Yuan et al., 2026) and LingBot-VA (Li et al., 2026), a frozen T5 encoder (Raffel et al., 2020) encodes the instructions. The two experts use a mixture-of-transformers layout with self-attention projections partially inherited from Wan2.2-5B (Wan et al., 2025). The Future and Action Experts contain 870M and 790M parameters, respectively. Together with the compactor and spatial encoder, they yield 2B trainable parameters. Both horizons are 32, $H_a = H_f = 32$.

Both branches share the same flow-matching formulation, with $\sigma = 5$ over 1000 levels. Only the Action Expert is integrated at inference, over 10 steps (Appendix A.1). Training runs AdamW at learning rate 10^{-4} ($2\times$ for the compactor and Future Expert), weight decay 10^{-2}, cosine annealing, bfloat16, gradient clipping 2.0, $\lambda_{\text{pred}} = 1$, EMA decay 0.996, and a global batch of 128 over eight GPUs for 30K steps. Every benchmark and ablation shares this recipe. Table 1a lists what differs.

4.2 EXPERIMENTAL SETTINGS

LIBERO and LIBERO-Plus. Following the standard LIBERO protocol (Liu et al., 2023), we evaluate all 40 tasks across the Spatial, Object, Goal, and Long suites, using 50 rollouts per task and the same closed-loop replanning protocol for every variant. For LIBERO-Plus (Fei et al., 2025), we report task-weighted success over 10,030 perturbed tasks. Comparison numbers are from the respective papers or from the evaluation of Zhang et al. (2026b).

RoboTwin. We evaluate each variant on all 50 official tasks, with 100 rollouts per task in each setting. *Clean2Clean* trains and evaluates on the clean set, whereas *Clean2Rand* evaluates clean-trained policies under joint randomization of layout, distractors, background, and lighting. Comparison numbers are from the benchmark leaderboard (Chen et al., 2025) and the respective papers.

Real robot. We evaluate *Fold the Towel* for long-horizon deformable-object manipulation, *Put Banana* for pick-and-place under adversarial object placements, and *Stack Three Bowls* for multi-stage rigid-object manipulation. The platform comprises two 6-degree-of-freedom arms with standard grippers and three RGB cameras providing one top and two wrist views. We use no depth input (Appendix D.1). Actions are represented as 14-D absolute joint positions (qpos). We collect 300 teleoperated demonstrations per task at 30 Hz, totaling approximately 4.3 hours. An automatic scene generator produces 50 seeded initial configurations per task, on which each policy is evaluated using real-time chunking (Black et al., 2025) (Appendix D.2).

Table 2: LIBERO and LIBERO-Plus success rates (%). WAM denotes explicit future modeling. Models that generate no future at deployment are marked with ‡. Bold and underlining mark the best and second best within each block throughout, including ties.

Method	Embodied Pretrain	WAM	LIBERO					LIBERO-Plus							
			Long	Goal	Object	Spatial	Avg.	Camera	Robot	Language	Light	Background	Noise	Layout	Total
Copper-Policy (ours)	×	✓‡	95.2	95.6	99.4	**98.8**	97.25	**78.67**	73.74	67.34	95.27	84.20	**94.44**	76.52	80.85
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	✓	×	92.4	98.0	98.2	**98.8**	96.9	75.4	77.5	85.6	**96.9**	94.6	89.7	**85.7**	**85.7**
Cosmos-Policy (Kim et al., 2026)	×	✓	**97.6**	98.2	**100.0**	98.1	98.5	75.8	63.3	81.7	96.5	88.9	92.7	82.2	82.2
ABot-M0 (Yang et al., 2026c)	✓	×	96.6	**99.0**	99.8	**98.8**	**98.6**	60.4	67.9	86.4	96.2	91.6	86.4	82.6	80.5
VLA-JEPA (Sun et al., 2026)	✓	✓‡	95.8	97.2	99.6	96.2	97.2	64.2	67.7	**88.1**	91.8	93.4	65.8	83.9	77.9
X-VLA (Zheng et al., 2026)	✓	×	**97.6**	97.8	98.6	98.2	98.1	23.4	**89.7**	75.7	88.2	**96.0**	62.7	71.8	71.4
Fast-WAM (Yuan et al., 2026)	×	✓‡	95.2	97.0	**100.0**	98.2	97.6	16.4	44.5	68.9	78.2	53.7	37.7	60.7	51.5
Copper-Policy Variants															
Copper-Policy			**95.2**	**95.6**	99.4	**98.8**	**97.25**	**78.67**	73.74	**67.34**	**95.27**	84.20	**94.44**	**76.52**	**80.85**
– w/o Joint-Embedding Prediction			91.2	95.0	**100.0**	97.4	95.90	76.17	**78.58**	53.94	93.52	**90.33**	91.63	76.26	79.11
– w/o Current-Frame Visual Features			90.6	95.4	99.0	96.8	95.45	76.49	71.23	52.77	95.01	74.26	81.32	72.72	74.11
– w/o World Input & Future Expert			90.8	92.4	98.8	97.2	94.80	66.79	67.23	52.18	95.18	88.20	87.07	74.49	74.56
– w/o Joint-Embedding Prediction & Visual Features			66.8	92.6	89.4	91.2	85.00	63.73	62.26	30.97	86.51	52.32	55.53	65.51	58.81



Configuration	Towel	Banana	Bowls	Avg.
$\pi_{0.5}$	95	94	94	94.3
Fast-WAM	**98**	60	94	84.0
Copper-Policy	93	**96**	**100**	**96.3**
w/o World Input & Future Expert	84	82	88	84.7
w/o Joint-Embedding Prediction	4	26	**100**	43.3
w/o Current-Frame Visual Features	2	46	14	20.7
w/o JE Pred. & Visual Features	0	4	0	1.3

Figure 3: **Real-robot evaluation.** The left panel shows four frames from one successful rollout per task, with target poses overlaid in the first column. The right panel reports scores that assign 1, 0.5, and 0 to full, half, and no success. $\pi_{0.5}$ uses embodied pretraining.

4.3 Main Results

4.3.1 LIBERO and LIBERO-Plus

Copper-Policy averages 97.25% on standard LIBERO. Its 80.85% LIBERO-Plus total exceeds several embodied-pretrained VLA baselines. Only Cosmos-Policy (82.2%), which fine-tunes Cosmos-Predict2 as its policy backbone, and large-scale embodied-pretrained $\pi_{0.5}$ (85.7%) score higher. Cosmos-Policy's small overall lead is driven by Language, as Copper-Policy has the higher pooled success rate over the other six categories. It leads Camera and Noise, ranks behind only X-VLA and $\pi_{0.5}$ on Robot among comparison methods, and remains strong on Light.

4.3.2 RoboTwin

Among methods without embodied pretraining, Copper-Policy achieves the highest average success rate at 41.91%, ahead of EventVLA at 40.65% and Fast-WAM at 39.85%. Its clean-set score is on par with $\pi_{0.5}$ (70.84% versus 70.70%). Under randomization, however, it remains substantially below $\pi_{0.5}$ (12.98% versus 46.00%).

4.3.3 Real-Robot Evaluation

Each policy uses the protocol and initial-scene seeds described above. Copper-Policy obtains the highest average, 96.3 against 94.3 for $\pi_{0.5}$ and 84.0 for Fast-WAM, and is best on both rigid-object tasks (Fig. 3). Removing the World input and Future Expert drops the average from 96.3 to 84.7. Fast-WAM performs best on towel folding at 98 but reaches 60 on banana placement, where it often does not attempt the grasp.

4.4 Efficiency

Predicting in a compact embedding space rather than generating video shortens the training sequence and permits smaller experts, putting the whole model at 2B trainable parameters. At a matched global batch of 128 on eight A100 GPUs, it uses less memory than $\pi_{0.5}$ and Fast-WAM

Table 3: RoboTwin success rates (%). Model info follows XPolicyLab (XPolicyLab et al., 2026). In the upper block, † marks the one entry trained without embodied pretraining; ‡ marks methods that generate no future at deployment.

Method	Model params.	Embodied Pretrain	WAM	RoboTwin		
				Clean	Random	Avg.
Comparison with methods using embodied pretraining						
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	3.3B	✓	×	70.70	**46.00**	**58.35**
Spatial Forcing (Li et al., 2025)	3.3B	✓	×	**77.20**	26.74	51.97
X-WAM (Guo et al., 2026)	5B	✓	✓	70.00	25.80	47.90
X-VLA (Zheng et al., 2026)	0.9B	✓	×	68.00	20.90	44.45
ABot-M0 (Yang et al., 2026c)	4.2B	✓	×	57.40	30.36	43.88
Copper-Policy (ours)	2B	×	✓‡	70.84	12.98	41.91†
Xiaomi Robotics-0 (Cai et al., 2026b)	4.7B	✓	×	62.90	18.20	40.55
GalaxeaVLA (Liu et al., 2026b)	3B	✓	×	62.70	12.72	37.71
RDT-1B (Liu et al., 2025)	1B	✓	×	34.50	13.72	24.11
Methods without embodied pretraining						
Copper-Policy (ours)	2B	×	✓‡	70.84	12.98	**41.91**
EventVLA (Yang et al., 2026b)	4.5B	×	×	65.60	**15.70**	40.65
Fast-WAM (Yuan et al., 2026)	6B	×	✓‡	**77.80**	1.90	39.85
AHA-WAM (Cai et al., 2026a)	7.2B	×	✓	64.30	3.20	33.75
DP3 (Ze et al., 2024)	255M	×	×	55.24	4.96	30.10
starVLA (Ye et al., 2026)	4B	×	×	46.52	3.16	24.84
ACT (Zhao et al., 2023)	80M	×	×	29.74	1.74	15.74
Copper-Policy Variants						
Copper-Policy				**70.84**	**12.98**	**41.91**
– w/o Joint-Embedding Prediction				48.16	9.38	28.77
– w/o Current-Frame Visual Features				67.82	10.22	39.02
– w/o World Input & Future Expert				55.26	10.22	32.74
– w/o Joint-Embedding Prediction & Visual Features				51.66	9.42	30.54

and trains about 2× and 6× faster. The same configuration completes 30K steps in 9.67 hours on eight consumer RTX 5090 GPUs with plain data parallelism. End-to-end latency is about 85 ms, against 75 ms for $\pi_{0.5}$ and 80 ms for Fast-WAM. This slightly higher latency reflects the frozen 300M V-JEPA encoder in the World Encoder, which runs at every step (Appendix B.2).

4.5 Controlled Comparisons

The compact World representation supports robustness to common closed-loop shifts. Camera, Robot, Light, and Noise test changes that commonly arise during closed-loop execution. Copper-Policy leads on Camera and Noise, remains strong on Light, and ranks behind only X-VLA and $\pi_{0.5}$ on Robot. Removing the World input and Future Expert lowers overall success from 80.85% to 74.56%, with clear losses on Camera, Robot, and Noise. Retaining the World input without joint-embedding prediction recovers much of the overall score, but still trails the full model on Camera and Noise while scoring higher on Robot. These comparisons support the compact World representation's value across common closed-loop shifts, while showing that predictive shaping benefits the shift categories differently.

Language, Background, and Layout expose the generalization gap. These categories test broader instruction and scene generalization, where several embodied-pretrained methods lead. Language is the clearest gap at 67.34%. Copper-Policy uses frozen T5 without a pretrained VLM for language–image understanding or large-scale embodied pretraining, consistent with this result. More diverse training may help, we will explore the representations these categories need.

Cross-benchmark evidence for the representation design. The variant without joint-embedding prediction stays close to the full model on LIBERO-Plus but falls from 41.91 to 28.77 on RoboTwin and from 96.3 to 43.3 on the real robot. Together, these ablations and the four closed-loop shift categories support the compact World representation's role in retaining task-relevant information across visual and robot changes. The larger losses on RoboTwin and the real robot suggest that this role matters more in physical control. Removing current-frame features lowers the LIBERO, LIBERO-Plus, and RoboTwin aggregate scores by at most 6.74 points, yet drops the real-robot average to 20.7, consistent with the need for current-frame detail in contact-rich execution (Section 4.6).

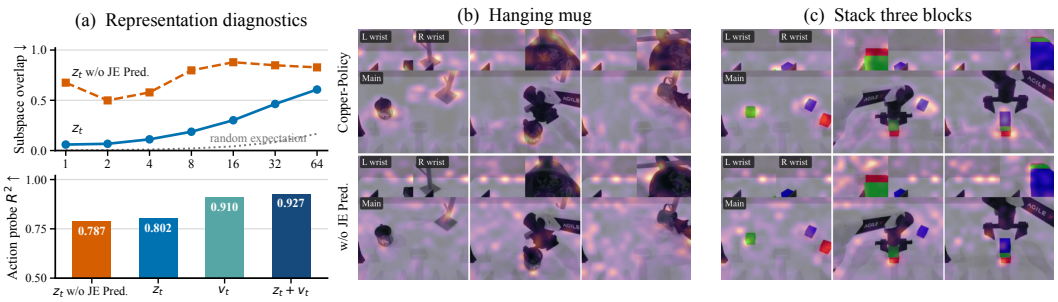


Figure 4: **Representation analysis.** (a) Perturbation–temporal subspace overlap across rank (top; dotted line: random expectation) and linear action-probe R^2 (bottom) on 100 RoboTwin episodes. (b–c) z_t attention across two RoboTwin episodes, comparing Copper-Policy (upper row) with *w/o Joint-Embedding Prediction* (lower row).

4.6 Representation Analysis

The comparisons above measure behavior. We now test the two representation-level predictions of Section 3.5. We use 100 offline RoboTwin episodes over four tasks and pair each frame with five deterministic image perturbations. We split by episode for subspace fitting and held-out projection evaluation (Appendix C).

The first prediction holds. Joint-embedding prediction reorganizes rather than removes nuisance information. Perturbation identity remains linearly decodable from z_t (0.998 balanced accuracy), yet the perturbation-change and temporal-change subspaces overlap far less with the objective across ranks (Fig. 4a, top). For the leading-k perturbation-induced (U_N) and temporal-change (U_T) subspaces, overlap is $\|U_N^\top U_T\|_F^2/k$. At rank 8, it is 0.187 versus 0.798 without the objective and 0.021 at random, with the same ordering on all four tasks (Appendix C). Across paired clean and perturbed views, the shaped z_t also changes less in magnitude (0.134 versus 0.344 mean normalized L_2 distance), while its attention remains on the manipulated object (Fig. 4b–c). Perturbations thus act along directions less aligned with task evolution, the representation-level counterpart of the Noise and Camera gains above.

The second prediction concerns the channels' division of information. On held-out clean episodes, a linear action probe yields R^2 of 0.802 from z_t, 0.910 from v_t, 0.927 from both, and 0.787 from z_t without joint-embedding prediction (Fig. 4a, bottom). Instantaneous action detail is more directly accessible from v_t, and real-robot rollouts show why combining the channels matters. Without v_t, z_t is the only visual input to both experts and remains as persistent as in the full model (0.904 versus 0.902), yet the average task score falls from 96.3 to 20.7. Under our compact budget, the temporally shaped World channel cannot replace the current-frame pathway for precise control (Appendix C).

5 Conclusion

Copper-Policy learns its predictive target space with the policy. Joint-embedding prediction shapes a compact World representation, and action decoding anchors it to control. An asymmetric prediction–control interface lets the World representation focus on task-relevant evolution while preserving current-frame detail for precise action decoding. Without embodied pretraining, it performs strongly across simulation and real-robot tasks, substantially improving on the two-stream Fast-WAM in LIBERO-Plus and real-robot scores while training faster with fewer parameters. Diagnostics show reduced overlap between perturbation-induced and task-driven variation in z_t and sustained attention to manipulated objects. Together with Camera, Robot, Light, and Noise results, this supports the representation's value under common closed-loop shifts.

Limitations. Strong Camera, Light, and Noise results despite the larger Language gap suggest the World representation is not narrowly tied to visual appearance. Instruction grounding remains limited, while Background and Layout reveal further scene-generalization gaps. Future work will learn representations that better capture task intent and scene structure across these settings.

ARTIFICIAL INTELLIGENCE (AI) USE STATEMENT

During manuscript preparation, we used generative AI tools for literature search, manuscript drafting and revision, final language polishing, and the generation of data-analysis and visualization code (e.g., matplotlib scripts). We did not use generative AI tools to generate datasets or data, or to run experiments. All AI-assisted text was reviewed and verified by the authors, and all AI-generated code was reviewed and approved by the authors before use. The authors remain responsible for the final manuscript, claims, data, code, and artifacts.

ETHICS STATEMENT

This work evaluates robot manipulation policies in simulation and on physical robot platforms. It does not involve human-subject experiments or the collection of personally identifiable information. Real-robot evaluations are limited to controlled manipulation tasks. Learned robot policies may fail under distribution shift and should undergo task-specific safety validation with appropriate human oversight before deployment.

REPRODUCIBILITY STATEMENT

Reproducibility is supported by the model and objective specification in Section 3, the evaluation protocols and ablation definitions in Section 4, and the implementation, target-construction, architecture, efficiency, and diagnostic details in the supplementary material (Appendices A, A.1, A.2, A.4, B.2, C.1 and D.3). The paper specifies the datasets, model configurations, evaluation protocols, and shared scene seeds where applicable; the supplementary tables provide per-task and outcome counts for recomputing the reported aggregates. Upon acceptance, we will release the complete training code.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, June 2023.

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.

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

Kevin Black, Allen Z Ren, Michael Equi, and Sergey Levine. Training-time action conditioning for efficient real-time chunking. *arXiv preprint arXiv:2512.05964*, 2025.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Maria Elisabeth Bechtle, Feryal Behbahani, Stephanie C.Y. Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando De Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 4603–4623. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/bruce24a.html.

Remi Cadene, Simon Alibert, Alexander Soare, Quentin Gallouedec, Adil Zouitine, Steven Palma, Pepijn Kooijmans, Michel Aractingi, Mustafa Shukor, Dana Aubakirova, Martino Russi, Francesco Capuano, Caroline Pascal, Jade Choghari, Khalil Meftah, Maxime Ellerbach, Jess Moss, and Thomas Wolf. Lerobot: State-of-the-art machine learning for real-world robotics in pytorch. https://github.com/huggingface/lerobot, 2024.

Jisong Cai, Long Ling, Shiwei Chu, Zhongshan Liu, Jiayue Kang, Zhixuan Liang, Wenjie Xu, Yinan Mao, Weinan Zhang, Xiaokang Yang, et al. Aha-wam: Asynchronous horizon-adaptive world-action modeling with observation-guided context routing. *arXiv preprint arXiv:2606.09811*, 2026a.

Rui Cai, Jun Guo, Xinze He, Piaopiao Jin, Jie Li, Bingxuan Lin, Futeng Liu, Wei Liu, Fei Ma, Kun Ma, et al. Xiaomi-robotics-0: An open-sourced vision-language-action model with real-time execution. *arXiv preprint arXiv:2602.12684*, 2026b.

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.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.026.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 2632–2652, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/0b408293619f725fd30162af057e531a-Paper-Conference.pdf.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 9156–9172. Curran Associates, Inc., 2023. doi: 10.52202/075280-0403. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1d5b9233ad716a43be5c0d3023cb82d0-Paper-Conference.pdf.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

Yixu Feng, Zinan Zhao, Yanxiang Ma, Chenghao Xia, Chengbin Du, Yunke Wang, and Chang Xu. See what matters: Differentiable grid sample pruning for generalizable vision-language-action model. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=HpIhfqaLwa.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Jun Guo, Qiwei Li, Peiyan Li, Zilong Chen, Nan Sun, Yifei Su, Heyun Wang, Yuan Zhang, Xinghang Li, and Huaping Liu. Unified 4d world action modeling from video priors with asynchronous denoising. *arXiv preprint arXiv:2604.26694*, 2026.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems 31*, pp. 2451–2463. Curran Associates, Inc., 2018. URL https://papers.nips.cc/paper/7512-recurrent-world-models-facilitate-policy-evolution. https://worldmodels.github.io.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, Apr 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-08744-2. URL https://doi.org/10.1038/s41586-025-08744-2.

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

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, et al. Cosmos policy: Fine-tuning video models for visuomotor control and planning. *arXiv preprint arXiv:2601.16163*, 2026.

Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to act from actionless videos through dense correspondences. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 40938–40958, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/b2d4051f03a7038a2771dfbbe5c7b54e-Paper-Conference.pdf.

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. *arXiv preprint arXiv:2510.12276*, 2025.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

Weixin Liang, LILI YU, Liang Luo, Srini Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen tau Yih, Luke Zettlemoyer, and Xi Victoria Lin. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=Nu6N69i8SB.

Yihan Lin, Jiawei He, Shifeng Bao, Chen Zhao, Yang Li, Xiaobo Wang, Yan Wang, Cheng Chi, and Jing Zhang. Jepa-wam: Learning vision-language-action policies with joint-embedding world modeling, 2026. URL https://arxiv.org/abs/2608.09381.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023. doi: 10.52202/075280-1939. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/8c3c666820ea055a77726d66fc7d447f-Paper-Datasets_and_Benchmarks.pdf.

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. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 29982–30009,

2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/49f80e4d2471ad4f2edf4f5f1ab62339-Paper-Conference.pdf.

Xiao Liu, Yuguang Yang, Xi Wang, Kai Jiang, Cheng Chi, Yong Xu, Wenchao Ding, Yilun Chen, and Yan Wang. Stagewam: Joint-embedding stage prediction for world-action models in robot manipulation, 2026a. URL https://arxiv.org/abs/2608.10780.

Yicheng Liu, Zibin Dong, Baijun Ye, Tianyuan Yuan, Tao Jiang, Anqi Yang, Shicheng Cao, Haonan Liu, Yue Sun, Zihan Guo, Xiao Liu, Dong Ke, Changxun Pan, Chenru Wu, Tailai Cheng, Xiaoshu Ren, Xinlei Zhang, Jianning Cui, Zijie Zhao, Haoyu Zhang, Kaiming Xu, Haodong Yang, Bowen Zhang, Jiahui Niu, Shaoting Zhu, Shiduo Zhang, and Hang Zhao. G0.5: One autoregressive stream for robot reasoning and action, 2026b. URL https://arxiv.org/abs/2608.11739.

Hao Luo, Wanpeng Zhang, Yicheng Feng, Sipeng Zheng, Haiweng Xu, Chaoyi Xu, Ziheng Xi, Yuhui Fu, and Zongqing Lu. Being-H0.7: A latent world-action model from egocentric videos. *arXiv preprint arXiv:2605.00078*, 2026.

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. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=YJ7o2wetJ2.

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

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 892–909. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/nair23a.html.

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

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.

Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 416–426. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/radosavovic23a.html.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

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

Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, Guangzhong Sun, Xin Jin, and Zhibo Chen. Vla-jepa: Enhancing vision-language-action model with latent world model. In *European Conference on Computer Vision*, pp. 478–497. Springer, 2026.

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

Community XPolicyLab, Tianxing Chen, Yue Chen, Tian Nian, Zijian Cai, Guangyu Chen, Wenwei Lin, Qiwei Liang, Peicheng Xiang, Kailun Su, et al. XPolicyLab: A unified standard and open ecosystem for robot policy evaluation and deployment. *arXiv preprint arXiv:2608.09892*, 2026.

Fan Yang, Yuting Su, Xiaobo Wang, Yuncheng You, Fugui Fan, Yuting Wu, Minghui Wu, Chenxu Zhao, JiaHong Ning, and Peiguang Jing. Lila-wam: Lightweight latent reasoning world-action model for robotic manipulation, 2026a. URL https://arxiv.org/abs/2608.03701.

Ganlin Yang, Zhangzheng Tu, Yuqiang Yang, Sitong Mao, Junyi Dong, Tianxing Chen, Jiaqi Peng, Jing Xiong, Jiafei Cao, Jifeng Dai, et al. Eventvla: Event-driven visual evidence memory for long-horizon vision-language-action policies. *arXiv preprint arXiv:2606.20092*, 2026b.

Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Dekang Qi, Junjin Xiao, Haoyun Liu, Ronghan Chen, Yuzhi Chen, Dongjie Huo, et al. Abot-m0: Vla foundation model for robotic manipulation with action manifold learning. *arXiv preprint arXiv:2602.11236*, 2026c.

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

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Se June Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, Lars Liden, Kimin Lee, Jianfeng Gao, Luke Zettlemoyer, Dieter Fox, and Minjoon Seo. Latent action pretraining from videos. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 28213–28239, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/45d74e190008c7bff2845ffc8e3facd3-Paper-Conference.pdf.

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. URL https://arxiv.org/abs/2603.16666.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D Diffusion Policy: Generalizable Visuomotor Policy Learning via Simple 3D Representations. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.067.

Jianjun Zhang, Jian Zhu, Taiyi Su, Chong Ma, Zitai Huang, Yi Xu, and Hanli Wang. Learning 4d geometric priors for inference-efficient world action models. *arXiv preprint arXiv:2607.05468*, 2026a.

Zhanguang Zhang, Zhiyuan Li, Behnam Rahmati, Rui Heng Yang, Yintao Ma, Amir Rasouli, Sajjad Pakdamansavoji, Yangzheng Wu, Lingfeng Zhang, Tongtong Cao, et al. Do world action models generalize better than vlas? a robustness study. *arXiv preprint arXiv:2603.22078*, 2026b.

Haoyu Zhao, Xingyue Zhao, Siteng Huang, Xin Li, Deli Zhao, and Zhongyu Li. Rynnworld-4d: 4d embodied world models for robotic manipulation. *arXiv preprint arXiv:2607.06559*, 2026.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.016.

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. In *International Conference on Learning Representations*, volume 2026, pp. 60580–60606, 2026.

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

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 79115–79135. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/zhou25t.html.

A Implementation Details

This section collects the training objective, target construction, representation modules, expert architecture, and token interface used by every experiment.

A.1 Flow-Matching Schedule and Future Targets

The interpolation coordinate is drawn as $u = \varphi(\rho)$ with $\rho \sim \mathcal{U}[0, 1]$ and shift $\sigma = 5.0$, over 1000 discrete training levels. We use the convention that $u = 0$ is the clean target and $u = 1$ pure noise, so the velocity target is $\epsilon - \tilde{z}$, with $\tilde{z}$ the pooled target sequence of Section 3.2. The same schedule and shift are used for LIBERO, RoboTwin, and the real robot. At test time the Future Expert stream runs once on the current tokens to prefill its per-layer keys and values; no future tokens are instantiated or sampled, and the Action Expert samples with 10 flow steps.

Future targets. The future horizon matches the action chunk, $H_f = H_a = 32$, and is represented by a smaller set of future target nodes. The frozen V-JEPA encoder produces spatial patch features for the current observation and for the future observations used to construct these targets. The online compactor consumes the current encoder output directly. For LIBERO, the future features are temporally downsampled to $N_f = 8$ target nodes; RoboTwin and the real robot use $N_f = 4$ sparse target nodes. Let $\bar{H}_j$ denote the encoder features associated with node j. The EMA compactor produces the stop-gradient target

$$\tilde{z}_j = \mathrm{sg}\big(C_{\mathrm{EMA}}(\bar{H}_j, e_s)\big), \qquad j = 1, \ldots, N_f.$$

Temporal downsampling does not reduce the number of compact tokens: each $\tilde{z}_j$ still contains K spatial compact tokens.

A.2 World Encoder Architecture

At time t, the World Encoder receives the synchronized multi-view observation $x_t = \{x_t^{(m)}\}_{m=1}^{M}$. A frozen 300M-parameter V-JEPA encoder maps the observation to spatial patch features. The same encoder produces the features used to construct future targets. We denote the current encoder output by

$$H_t^{\mathrm{VE}} \in \mathbb{R}^{P \times 1024}.$$

The views are concatenated in the fixed camera order and $P = \sum_m P_m$ is the total number of spatial patches. The current encoder output is consumed directly by the online compactor, while future encoder outputs are grouped and mean-pooled before entering the EMA target compactor as described above. These features are detached from the policy loss. The compact World pathway consumes only the spatial patch features, preserving the patch-to-camera correspondence needed by the compactor while keeping the World input independent of the Action Expert.

The learned compactor C_{θ_C} maps the dense patch sequence to a small set of abstract tokens,

$$z_t = C_{\theta_C}(H_t^{\mathrm{VE}}, e_s) \in \mathbb{R}^{K \times d_z}, \qquad d_z = 384.$$

It is a four-block resampler with six attention heads, head dimension 64, an MLP ratio of 4, and a final layer normalization. The number of compact tokens is determined by the input resolution, patch size, and band size rather than set independently. With the multi-view layout used for LIBERO, this yields $K = 28$; the T-shaped visual layout used for RoboTwin and the real robot yields $K = 36$. The learnable queries are typed as horizontal-band, vertical-band, or global queries. A horizontal (vertical) query can read only the corresponding two-row (two-column) patch band in each camera view; global queries, when present, can read all views. Thus the compactor retains an explicit mapping between each latent slot and image regions. The compactor output is projected to the 1280-dimensional expert width before it enters the joint transformer.

With task conditioning, the instruction embedding is used only to form the compactor's patch keys: patch features are projected to the compactor width, the T5 context is projected to the same width, and a patch-to-instruction cross-attention produces a task-conditioned patch context. The key network receives the concatenation of the patch projection, this context, and their elementwise product;

values are obtained from the patch projection. The compact queries then read these keys and values through four residual attention–MLP blocks. This conditioning makes the World bottleneck task-aware without creating a separate visual token stream in the expert.

The same compactor is maintained as an exponential-moving-average target encoder with decay 0.996. The EMA copy supplies stop-gradient future targets, whereas the online copy supplies z_t to the policy. The World pathway is compact and jointly shaped by joint-embedding prediction and action decoding, while the current-frame pathway below remains available for instantaneous spatial evidence. The computational graph and token routing of this module are shown in Fig. A1.

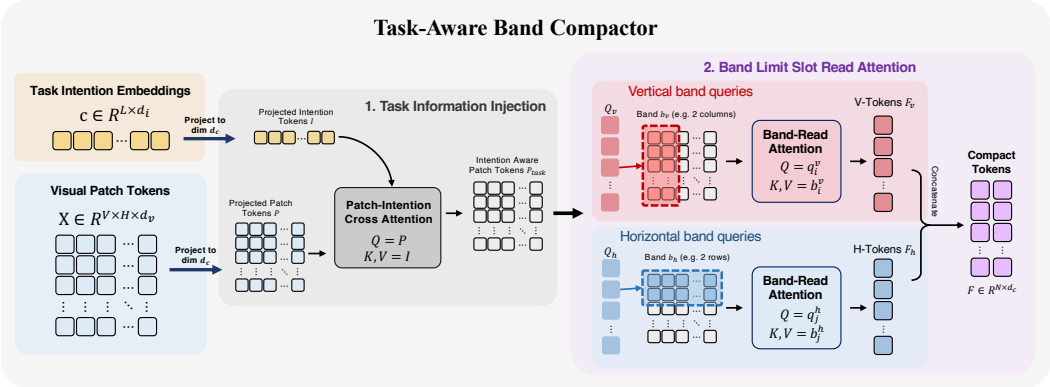


Figure A1: **Task-aware band compactor.** Task-intention embeddings first condition the visual patch keys through patch–intention cross-attention. Horizontal and vertical band queries then read restricted patch bands and concatenate their outputs into the compact representation.

A.3 Current-Frame Vision Architecture

The current-frame pathway is computed from the same observation x_t, but it is not forced through the World bottleneck. Each camera view is processed independently by a trainable spatial encoder initialized from DINOv2-L with registers. Its architecture has hidden width 1024, 24 transformer layers, 16 attention heads, patch size 14, and four register tokens; only the spatial patch tokens are retained. For a view with patch grid $h_m \times w_m$, the spatial output is reshaped to that grid and the views are concatenated in camera order:

$$S_t = \mathrm{DINO}_{\theta_V}(x_t) \in \mathbb{R}^{P \times 1024}.$$

Unlike the frozen V-JEPA source used by the compactor, the spatial encoder and sampler parameters θ_V are learnable.

The current-frame sampler uses a coordinate head that predicts L normalized two-dimensional locations from the pooled spatial features. For multi-view inputs, one shared MLP maps the pooled multi-view feature to $2L$ sigmoid outputs; when views are processed separately, the same operation is applied to each view. The predicted coordinates are used by bilinear `grid_sample` on the patch feature map. The sampled features are linearly projected to the 1280-dimensional expert width and receive a learned coordinate embedding, producing

$$v_t = G_{\theta_V}(S_t) \in \mathbb{R}^{L \times 1280}.$$

The sampler uses $L = 16$ tokens per view on LIBERO and $L = 32$ tokens in total across the three-view RoboTwin and real-robot runs. In ablations that retain every spatial patch, the patch tokens are projected directly. In both cases the current-frame tokens are concatenated after the abstract World tokens, so the policy receives the ordered prefix $[z_t; v_t]$.

The two pathways therefore differ in both supervision and information bottleneck. z_t is a small, temporally structured representation trained to support future prediction; v_t is a reactive, spatially grounded representation of the current image. The structured mask lets the Action Expert read both streams, while the Future Expert reads the World stream and does not directly read v_t (Appendix A.4).

A.4 Expert Architecture and Attention Mask

The two experts are streams of one joint self-attention transformer. From Wan2.2-5B (Wan et al., 2025) we transfer each block's self-attention projections, query/key RMSNorm weights, and AdaLN modulation, together with compatible text/time conditioning projections. Because the Wan blocks are width 3072 while ours are width 1280, shape-mismatched transferred matrices are resized by bilinear interpolation; one-dimensional parameters use one-dimensional interpolation. Cross-attention and feed-forward layers are trained from scratch. Both streams run 30 layers of width 1280 with 10 heads of dimension 128, and each layer cross-attends to the T5 instruction embeddings of dimension 4096. Their input and output projection dimensions and feed-forward dimensions differ; the Future Expert uses a feed-forward ratio of $14336/3072 \approx 4.67$ and the Action Expert uses 4, contributing to their respective 870M and 790M parameter counts. The structured mask of Sections 3.3 and 3.4 is applied inside the joint self-attention, so the streams exchange information only where it allows. Written out by query group, the World tokens z_t and the current-frame tokens v_t attend to each other and to themselves, the future queries attend to z_t and to other future tokens, and the action queries attend to z_t, v_t, and other action tokens. The future queries therefore never read v_t directly and v_t is never a prediction target, while the prefix itself is fused, so the predictive gradient can reach the current-frame pathway through the z_t rows.

A.5 Transformer Sequence Composition

Table A1: Transformer sequence composition per sample. For RoboTwin and real-robot runs, the T-shaped visual layout gives 68 current tokens from 32 visual and 36 World tokens; four future target nodes give 144. The multi-view LIBERO layout yields $K = 28$. Of the current tokens the Future Expert sees only the K World tokens, since the mask of Section 3.3 excludes v_t. Each method uses 32 action tokens; a dash marks a method with no future branch.

Method	Current vis./world	Future	Action	Total
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	768	–	32	800
Fast-WAM (Yuan et al., 2026)	120	240	32	392
Copper-Policy	**68**	**144**	32	**244**

B Efficiency and Accessibility

B.1 Design Scope

Compact prediction targets shorten the training sequence. Copper-Policy has 2B trainable parameters, and the complete 30K-step run uses plain data parallelism on eight consumer GPUs. The matched component interventions and evaluations across LIBERO, LIBERO-Plus, RoboTwin, and the real robot test the complete formulation.

B.2 Efficiency Measurements

Table A2: Training and inference efficiency.

Method	Trainable params.	Training hardware	Parallelism	Peak mem. / GPU ↓	Relative training time ↓	Samples / s ↑	Est. time (30K steps) ↓	RTX 5090 inference ↓
Fast-WAM (Yuan et al., 2026)	6B	8× A100 80GB	FSDP Fully Shard	62.42 GB	6.13×	14.36	74.3 h	∼80 ms
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	3.3B	8× A100 80GB	FSDP Fully Shard	38.65 GB	1.96×	44.91	23.7 h	**∼75 ms**
Copper-Policy (ours)	**2B**	8× A100 80GB	DDP	31.83 GB	1.00×	88.0	12.1 h	∼85 ms
Copper-Policy under alternative memory-sharding strategies (8× A100 80GB)								
Copper-Policy	2B	8× A100 80GB	DDP	31.83 GB	1.00×	88.0	12.1 h	∼85 ms
Copper-Policy	2B	8× A100 80GB	DDP + Optimizer Shard	24.53 GB	1.08×	81.8	13.0 h	∼85 ms
Copper-Policy	2B	8× A100 80GB	FSDP Fully Shard	**21.24 GB**	1.50×	58.56	18.2 h	∼85 ms
Copper-Policy (ours)	**2B**	**8× RTX 5090 32GB**	DDP	31.83 GB	**0.80×**	**110.34**	**9.67 h**	∼85 ms

Table A2 reports training cost at global batch 128, normalized to Copper-Policy under DDP on 8×A100 (1.454 s/step). Peak memory is the reserved memory per GPU in GB (10^9 bytes). *FSDP*

Fully Shard covers ZeRO-3, FSDP, and FSDP2 full-shard, while *DDP + Optimizer Shard* denotes ZeRO-1. Estimated time assumes 30K steps, and inference is measured end to end.

The middle block of Table A2 measures the memory and throughput effects of sharding within one implementation. Copper-Policy trains under plain DDP with 31.83 GB per GPU. Optimizer sharding reduces memory to 24.53 GB and raises training time to 1.08×. Full sharding uses 21.24 GB at 1.50× the training time. Both baselines require more memory even with full sharding. $\pi_{0.5}$ uses 38.65 GB and Fast-WAM uses 62.42 GB. Part of the measured speed difference therefore comes from the sharding overhead that Copper-Policy avoids. Its 31.83 GB memory use also fits the 32 GB available on an RTX 5090, allowing the same setup to train on eight consumer GPUs in 9.67 hours.

These are operational cost references rather than a controlled benchmark. The methods differ in distributed-training strategy and implementation, so the relative-time column gives wall-clock multiples rather than isolating one factor. The main contributors are sequence length (Appendix A.5), model size, and the sharding overhead Copper-Policy avoids. Among the baselines $\pi_{0.5}$ is the smaller, at 3.3B against Fast-WAM's 6B. Copper-Policy trains 1.96× faster than it and runs unsharded. Inference is reported separately because training throughput does not determine it. With compiled inference, latency is about 85 ms for Copper-Policy, 80 ms for Fast-WAM, and 75 ms for $\pi_{0.5}$. The frozen 300M V-JEPA encoder in our World Encoder runs at every step.

C Representation Analysis

C.1 Representation Diagnostics Protocols

All representation diagnostics are read-only evaluations of frozen checkpoints. We run the model in evaluation mode with gradients disabled and use the same synchronized observations and language conditioning as in policy inference. For temporal metrics, the episode is the statistical unit; the larger perturbation analysis splits data by episode. For the World representation, the exported token and channel dimensions are flattened before temporal metrics are computed. For the current-frame pathway, the exported representation is the mean of its spatial tokens. Episode-level averaging gives each episode equal weight.

Temporal stability. For each episode we export the online World representation z_t and, for the full model, the mean-pooled current-frame representation v_t. At horizons $\Delta \in \{1, 2, 4, 8, 16\}$ we compute cosine similarity and Euclidean displacement between unit-normalized representations,

$$c_\Delta = \frac{1}{T-\Delta} \sum_t \hat{r}_t^\top \hat{r}_{t+\Delta}, \qquad d_\Delta = \frac{1}{T-\Delta} \sum_t \|\hat{r}_{t+\Delta} - \hat{r}_t\|_2 .$$

We average the resulting episode-level values and report 95% confidence intervals across episodes. The main temporal comparison uses 40 LIBERO episodes, 20 RoboTwin episodes, and 18 real-robot episodes. RoboTwin is subsampled every fourth frame, so the same index separation spans a longer physical interval there; its absolute values are therefore not compared numerically with LIBERO or the real robot.

Perturbation geometry and information probes. The larger perturbation analysis uses 100 RoboTwin episodes: 25 episodes from each of four tasks, with 16 uniformly spaced valid frames per episode. Each clean frame is paired with five deterministic visual perturbations—camera translation, illumination change, global color cast, Gaussian sensor noise, and blur—yielding 1,600 clean states and 8,000 paired examples. Splits are made by episode, never by frame. Geometry and probe results are averaged over five episode-level splits; paired clean/perturbed metrics are averaged over all 8,000 pairs. For geometry at rank $k = 8$, we fit the leading perturbation-change subspace U_N and adjacent-frame temporal-change subspace U_T on training episodes. Their overlap is computed directly from the fitted bases, without held-out samples. Separately, we evaluate the fraction of held-out perturbation energy projected onto the temporal subspace. The two quantities are

$$\frac{\left\|U_N^\top U_T\right\|_F^2}{k} \quad \text{and} \quad \frac{\|U_T U_T^\top \delta_N\|_2^2}{\|\delta_N\|_2^2}$$

Here δ_N is a perturbation-induced change from a held-out episode. A random subspace of the same rank is the reference. Linear nuisance and action probes are fit on the training episodes only; the

former predicts perturbation identity, while the latter predicts the action chunk. For v_t, we mean-pool current-frame tokens before probing. We also report the paired clean/perturbed L_2 distance and cosine similarity.

Qualitative protocol. For the z_t attention pages, the full model and *w/o Joint-Embedding Prediction* receive identical frames, task instructions, camera ordering, and rendering schedules. We render six evenly spaced steps per selected episode and preserve the native multi-view layout.

C.2 Temporal Structure and Channel Complementarity

The comparisons so far measure behavior. Because the formulation is a claim about a representation, we also inspect z_t directly. The temporal analysis follows the protocol in Appendix C.1, with the episode rather than the frame as the statistical unit.

Table A3: Temporal structure of the World representation, averaged over episodes. Cosine similarity and displacement are computed between unit-normalized representations 16 sampled frames apart; RoboTwin samples every fourth frame, so its separation spans more time and the three domains are not directly comparable. Bold and underlining mark the best and second best in each column.

Variant	LIBERO ($n = 40$)		RoboTwin ($n = 20$)		Real robot ($n = 18$)	
	cosine ↑	disp. ↓	cosine ↑	disp. ↓	cosine ↑	disp. ↓
Copper-Policy	**0.794**	**0.634**	**0.913**	**0.413**	<u>0.902</u>	<u>0.434</u>
– w/o Joint-Embedding Prediction	0.684	0.778	0.729	0.724	0.756	0.684
– w/o Current-Frame Visual Features	<u>0.778</u>	<u>0.660</u>	<u>0.890</u>	<u>0.463</u>	**0.904**	**0.429**
– w/o Joint-Embedding Prediction & Visual Features	0.665	0.809	0.567	0.921	0.773	0.658

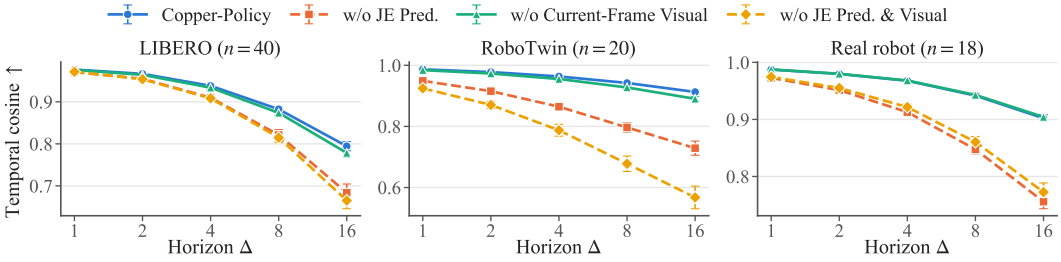


Figure A2: Cosine similarity of unit-normalized z_t across horizon Δ on LIBERO, RoboTwin, and the real robot, with 95% confidence intervals over episodes.

Joint-embedding prediction organizes the World representation in time. Table A3 reports every variant at $\Delta = 16$, and Fig. A2 traces cosine similarity across horizons. Removing joint-embedding prediction lowers cosine similarity in every domain, from 0.794 to 0.684 on LIBERO, from 0.902 to 0.756 on the real robot, and from 0.913 to 0.729 on RoboTwin, and raises displacement accordingly; without the joint-embedding objective the compactor still produces features, but they change faster over the episode. The paired comparison without v_t keeps the ordering (0.778 versus 0.665, 0.904 versus 0.773, and 0.890 versus 0.567). Figures A6 to A8 show the corresponding spatial pattern over the complete set of z_t attention sequences. Across the three domains, joint-embedding prediction keeps z_t focused on manipulated objects and other task-relevant regions as an episode proceeds, whereas without the objective the maps more often retain maxima on background surfaces and lack a stable task-relevant center.

The diagnostics accompany a behavioral reversal. As noted in Section 4.5, *w/o Joint-Embedding Prediction* scores below *w/o World Input & Future Expert* on RoboTwin (28.77% versus 32.74%) and on the real robot (43.3 versus 84.7), despite retaining the World pathway. The temporal diagnostics show that its z_t drifts faster over episodes, while the attention maps show that it lacks a stable task-relevant focus. Together, these observations characterize the unshaped World signal accompanying the lower control scores.

The two pathways differ in temporal stability as well as spatial focus. Removing the current-frame pathway perturbs z_t far less than removing joint-embedding prediction does: cosine similarity moves from 0.794 to 0.778 on LIBERO and from 0.913 to 0.890 on RoboTwin, and on the real robot the two are indistinguishable (0.902 versus 0.904). Removing both joint-embedding prediction and the current-frame pathway is the largest drop on RoboTwin, though on the real robot it is no lower than removing joint-embedding prediction alone. Within the full model, z_t is also more stable than the pooled current-frame tokens v_t at the same separation, on LIBERO (0.794 versus 0.672), on the real robot (0.902 versus 0.827), and most clearly on RoboTwin (0.913 versus 0.245). The current-frame tokens therefore change faster over an episode than the World representation does.

Perturbation geometry and channel probes. The perturbation-geometry and channel-probe results follow the larger protocol in Appendix C.1, using frozen inference and episode-level splits.

Table A4: Representation diagnostics on 100 RoboTwin episodes. Geometry subspaces are fit on training episodes; overlap compares the fitted subspaces, while projected perturbation energy and probe scores are evaluated on held-out episodes. The random-subspace overlap is 0.021 and the random energy 0.021–0.023. Paired columns average 8,000 clean/perturbed pairs. The action probe is a linear readout of the action chunk on held-out clean episodes. Standard deviations over five episode-level splits are at most 0.006 for the geometry columns and 0.021 for R^2.

	Geometry at rank 8		Paired perturbation		Action probe
Representation	overlap ↓	pert. in temp. ↓	norm. L_2 ↓	cosine ↑	R^2 ↑
Copper z_t	**0.187**	**0.140**	**0.134**	**0.990**	0.802
– w/o Joint-Embedding Prediction	<u>0.798</u>	<u>0.807</u>	0.344	0.932	0.787
Current-frame tokens v_t	–	–	<u>0.171</u>	<u>0.969</u>	<u>0.910</u>
Dense V-JEPA features	–	–	0.428	0.904	0.831
Token-matched V-JEPA features	–	–	0.462	0.891	0.803
z_t and v_t together	–	–	–	–	**0.927**

The same overlap ordering holds separately on each of the four tasks.

Perturbation sensitivity is reorganized rather than removed. Perturbation identity remains linearly decodable from z_t at 0.998 balanced accuracy, compared with 0.997 without joint-embedding prediction and 0.893 from v_t. The ratio of perturbation change to adjacent temporal change is 1.55 (95% episode-bootstrap interval 1.46–1.65) for z_t, 1.12 (1.05–1.20) without joint-embedding prediction, and 0.219 (0.204–0.234) for v_t: z_t is smoother in absolute terms, but not selectively invariant to these perturbations. The probes measure linear readout, and the geometry analysis covers the five synthetic perturbations listed above.

C.3 Attention-Map Generation

For the z_t attention shown in the main text and the per-episode pages below, we capture the attention probabilities in the final task-conditioned compactor read block. If A_{hqp} denotes the softmax probability from compact query q and head h to input patch p, the scalar patch score is

$$m_p^Z = \frac{1}{HQ} \sum_{h=1}^{H} \sum_{q=1}^{Q} A_{hqp}.$$

The patch sequence is then split according to the original camera grids, $m_p^Z \mapsto m^{Z,(m)} \in \mathbb{R}^{h_m \times w_m}$. The resulting z_t attention is the final-compactor read attention averaged over heads and compact queries.

For the current-frame representation map, we compute a post-hoc source-patch response from the tokens produced by the coordinate-based bilinear sampler. Let q_ℓ be a current-frame token and k_p the projected source-patch feature used by the sampler. With temperature $\tau = 0.10$, we compute

$$a_{\ell p} = \mathrm{softmax}_p\left(\frac{\mathrm{norm}(q_\ell)^\top \mathrm{norm}(k_p)}{\tau}\right), \qquad m_p^V = \frac{1}{L} \sum_{\ell=1}^{L} a_{\ell p}.$$

m^V is the *current-frame representation map*: it measures which source patches are most similar to the information carried by the sampled v_t tokens.

For rendering, each per-view map is reshaped to its patch grid, resized to the corresponding input image with bicubic interpolation, and optionally smoothed with a Gaussian kernel of $\sigma = 6$ pixels. The map is min–max normalized independently within each view and overlaid with opacity 0.45 on the min–max normalized RGB frame with opacity 0.55. We use the MAGMA colormap for m^Z and VIRIDIS for m^V; color intensity is comparable within a map family and frame. Multi-camera panels preserve the model's camera order; for three-view RoboTwin observations the display uses the T-shaped canvas with the two wrist views above the front view.

D Real-Robot Evaluation Details

D.1 Experimental Platform

Our experiments use an AgileX Cobot equipped with two 6-degree-of-freedom Piper arms and standard grippers (Fig. A3). Robot communication, state acquisition, and joint-position command execution use the SDK provided by AgileX. Three Intel Corp. RealSense D435 cameras provide multi-view observations, of which the policy uses only the RGB streams. The action space comprises 14-D absolute joint positions (qpos). Teleoperated demonstrations are stored in the LeRobot dataset format (Cadene et al., 2024).



Figure A3: Real-robot experimental platform. The AgileX Cobot is equipped with two Piper arms and standard grippers (left); three Intel Corp. RealSense D435 cameras provide one top and two wrist views (right).

D.2 Real-Time Chunking

Evaluation scenes are produced by an automatic generator rather than placed by hand. We fit the distribution of object positions and orientations in the training set, sample from it with jitter, and hold the seeds fixed. Every policy therefore faces the same 50 initial scenes per task.

All evaluated policies use real-time chunking (RTC) (Black et al., 2025) at deployment. Copper-Policy also simulates RTC delay during training by conditioning the Action Expert on a committed action prefix whose length is sampled from 0 to 6 steps. For real-robot deployment, asynchronous RTC executes the action chunk $a^+ := a_{t:t+H_a-1}$ in closed loop: while the robot executes a committed prefix, the policy server infers the remaining part of the next chunk from the latest multi-view observation. After an initial chunk is committed, each replanning step preserves a committed prefix whose length is determined by the estimated end-to-end inference delay, while the Action Expert regenerates the remaining, uncommitted portion; RTC then replans at 30 Hz. This overlap

between execution and inference maintains continuous action execution despite inference latency. The committed-prefix mechanism is specific to real-robot execution and is distinct from the action tokens the Action Expert otherwise generates during training and simulation evaluation.

D.3 Protocol and Outcome Counts

All policies, including $\pi_{0.5}$ and Fast-WAM, follow the same evaluation protocol and scene setup. They are trained or fine-tuned on the same 300 demonstrations per task, deployed on the same platform with the same cameras and action space, and evaluated on 50 trials per task from shared initial-scene seeds. Table A5 reports full-success, half-success, and failure counts, scored as 1, 0.5, and 0. Trial outcomes are assigned by manual annotation of the recorded rollouts. A trial is a full success when the task goal is reached, a half success when the goal is only partially reached, and a failure otherwise. Half success is awarded only on towel folding, when the fold is completed but a visible crease or corner remains. Banana placement and bowl stacking admit no partial credit. The per-task counts make the outcome distribution explicit so that scores can be recomputed under another weighting. The reported average uses all 150 trials per policy variant.

Table A5: Per-task outcome counts behind the aggregate real-robot scores.

Variant	Full / half / failure counts			Avg. score
	Fold Towel	Put Banana	Stack Bowls	
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	47/1/2	47/0/3	47/0/3	94.3
Fast-WAM (Yuan et al., 2026)	49/0/1	30/0/20	47/0/3	84.0
Copper-Policy	44/5/1	48/0/2	50/0/0	96.3
w/o World Input & Future Expert	41/2/7	41/0/9	44/0/6	84.7
w/o Joint-Embedding Prediction	1/2/47	13/0/37	50/0/0	43.3
w/o Current-Frame Visual Features	1/0/49	23/0/27	7/0/43	20.7
w/o Joint-Embedding Prediction & Visual Features	0/0/50	2/0/48	0/0/50	1.3

Additional qualitative observations. Successful Copper-Policy rollouts sometimes reach the task goals along trajectories visibly different from the teleoperated demonstrations. Fast-WAM often does not attempt the grasp in banana placement. In an additional unscored check outside the 150 reported trials, rotating the banana toward orientations present in the demonstrations recovered some grasp attempts.

E Additional RoboTwin Results

E.1 Per-Task Clean and Randomized Evaluation

Table A6 lists all 50 RoboTwin tasks for every ablation, under both the clean and the randomized setting, at 100 episodes per task. The column averages reproduce the aggregates of Table 3.

Table A6: **Per-task RoboTwin success rates (%).** All 50 tasks, 100 episodes each, under the clean and randomized settings. In every row and in the average, bold and underlining mark the best and second best, ties included; rows on which all five variants agree are left unmarked. *w/o World & FE* is *w/o World Input & Future Expert*; *w/o Visual Feat.* is *w/o Current-Frame Visual Features*.

Task	Copper-Policy		w/o JE Pred.		w/o Visual Feat.		w/o World & FE		w/o JE & Visual	
	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.
adjust bottle	**100**	35	**100**	40	**100**	0	**100**	**55**	98	1
beat block hammer	84	7	42	6	**86**	4	66	**11**	26	7
blocks ranking rgb	**82**	1	45	1	70	1	70	1	45	1
blocks ranking size	**46**	0	12	0	27	0	33	0	19	0
click alarmclock	**97**	32	77	17	93	**34**	94	15	89	13
click bell	97	14	71	12	96	**24**	**99**	8	79	13
dump bin bigbin	93	69	88	52	88	68	**94**	**70**	89	65
grab roller	**99**	**40**	86	25	98	9	84	32	94	12
handover block	**76**	0	41	0	70	0	39	**1**	41	0
handover mic	98	1	87	**3**	97	2	**100**	2	82	2
hanging mug	15	**2**	14	0	**25**	**2**	17	1	4	**2**
lift pot	**89**	**1**	37	0	84	**1**	38	0	57	0
move can pot	**59**	1	23	2	58	1	30	**8**	31	1
move pillbottle pad	**76**	**1**	21	0	67	0	26	0	38	0
move playingcard away	91	13	82	12	86	4	**92**	**18**	65	9
move stapler pad	**21**	0	2	0	17	0	8	0	4	**1**
open laptop	**95**	**14**	83	9	84	0	89	9	77	7
open microwave	49	7	46	18	**79**	1	37	13	20	**24**
pick diverse bottles	54	**12**	23	11	**63**	9	24	5	43	8
pick dual bottles	76	**16**	30	10	**90**	**16**	24	4	61	7
place a2b left	**40**	**4**	25	1	28	2	31	3	19	1
place a2b right	33	**5**	24	1	**40**	1	23	2	15	3
place bread basket	68	**35**	48	21	65	33	46	27	**69**	28
place bread skillet	**72**	**3**	38	0	**72**	2	34	2	48	2
place burger fries	**97**	3	81	2	91	**5**	75	3	82	4
place can basket	64	3	32	2	**70**	**6**	66	2	50	2
place cans plasticbox	**92**	**4**	29	**4**	59	**4**	35	0	55	3
place container plate	**99**	**48**	86	14	97	32	92	13	92	38
place dual shoes	**45**	**8**	7	0	43	1	13	3	30	0
place empty cup	**95**	**11**	44	2	92	8	71	3	62	9
place fan	**54**	**12**	21	1	49	1	30	3	25	4
place mouse pad	**36**	0	5	0	24	0	8	0	8	0
place object basket	**86**	**7**	85	**7**	76	6	78	3	74	2
place object scale	**52**	**1**	18	**1**	34	**1**	34	**1**	15	0
place object stand	**82**	10	65	11	76	7	75	17	60	**24**
place phone stand	70	**4**	38	**4**	**71**	2	45	1	36	2
place shoe	**79**	6	33	5	68	4	49	4	50	**10**
press stapler	**98**	14	69	**24**	79	15	85	18	87	23
put bottles dustbin	41	7	37	12	51	**14**	**53**	10	52	8
put object cabinet	40	6	36	**8**	37	7	42	2	**47**	0
rotate qrcode	76	**3**	57	1	**78**	1	68	2	58	1
scan object	44	0	29	**2**	**53**	1	28	0	45	0
shake bottle	99	**60**	98	41	**100**	49	99	48	**100**	31
shake bottle horizontally	**100**	**60**	**100**	44	**100**	53	**100**	54	98	33
stack blocks three	**56**	**3**	31	0	53	0	28	0	5	0
stack blocks two	**95**	3	64	2	88	3	88	**7**	64	1
stack bowls three	**76**	**39**	70	15	68	38	66	15	52	23
stack bowls two	89	16	94	15	**95**	27	93	9	89	**37**
stamp seal	**32**	0	12	0	18	**1**	18	0	10	0
turn switch	35	8	22	**11**	**38**	**11**	26	6	24	9
Average	**70.84**	**12.98**	48.16	9.38	67.82	10.22	55.26	10.22	51.66	9.42

F Accompanying Videos and Setup Materials

The accompanying archive contains real-robot evaluation and demonstration videos. It also includes additional videos visualizing the learned representations.

G REPRESENTATION VISUALIZATIONS

Figures A4 and A5 trace both representation maps through one LIBERO episode and one RoboTwin episode. Frames are sampled uniformly over each episode and ordered from top to bottom in the left column, followed by the right column. The subsequent pages report the complete z_t attention sequences used in the analysis: seven LIBERO episodes spanning all four suites, one episode from each of the four analyzed RoboTwin tasks, and one episode from each of the three real-robot tasks. Each block compares Copper-Policy (upper row) with *w/o Joint-Embedding Prediction* (lower row) at six evenly spaced steps of the same episode, using every camera received by the policy and identical frames for both variants.

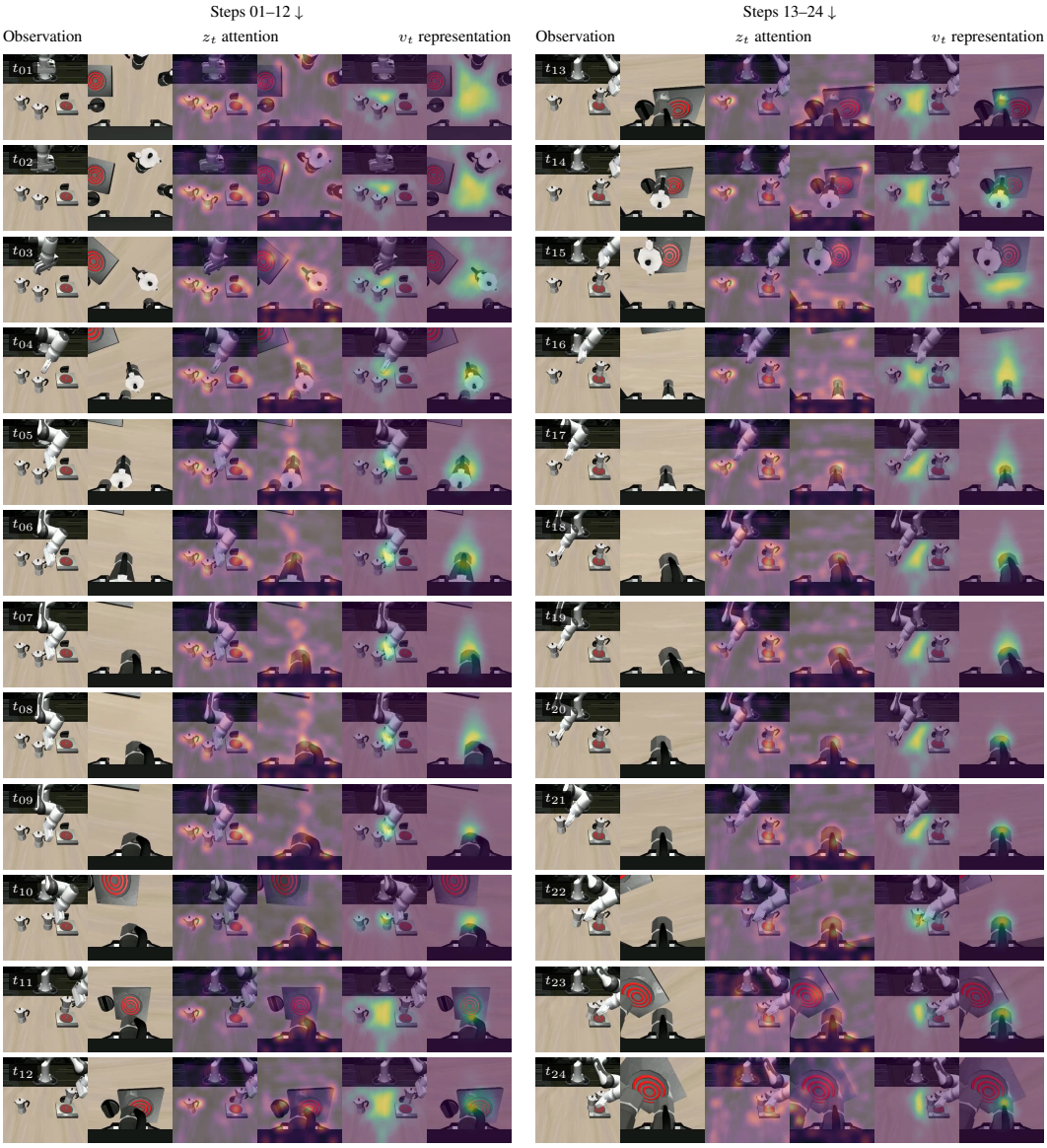


Figure A4: **LIBERO episode: put both moka pots on the stove.** Uniformly sampled observations with the corresponding z_t attention and v_t representation maps.

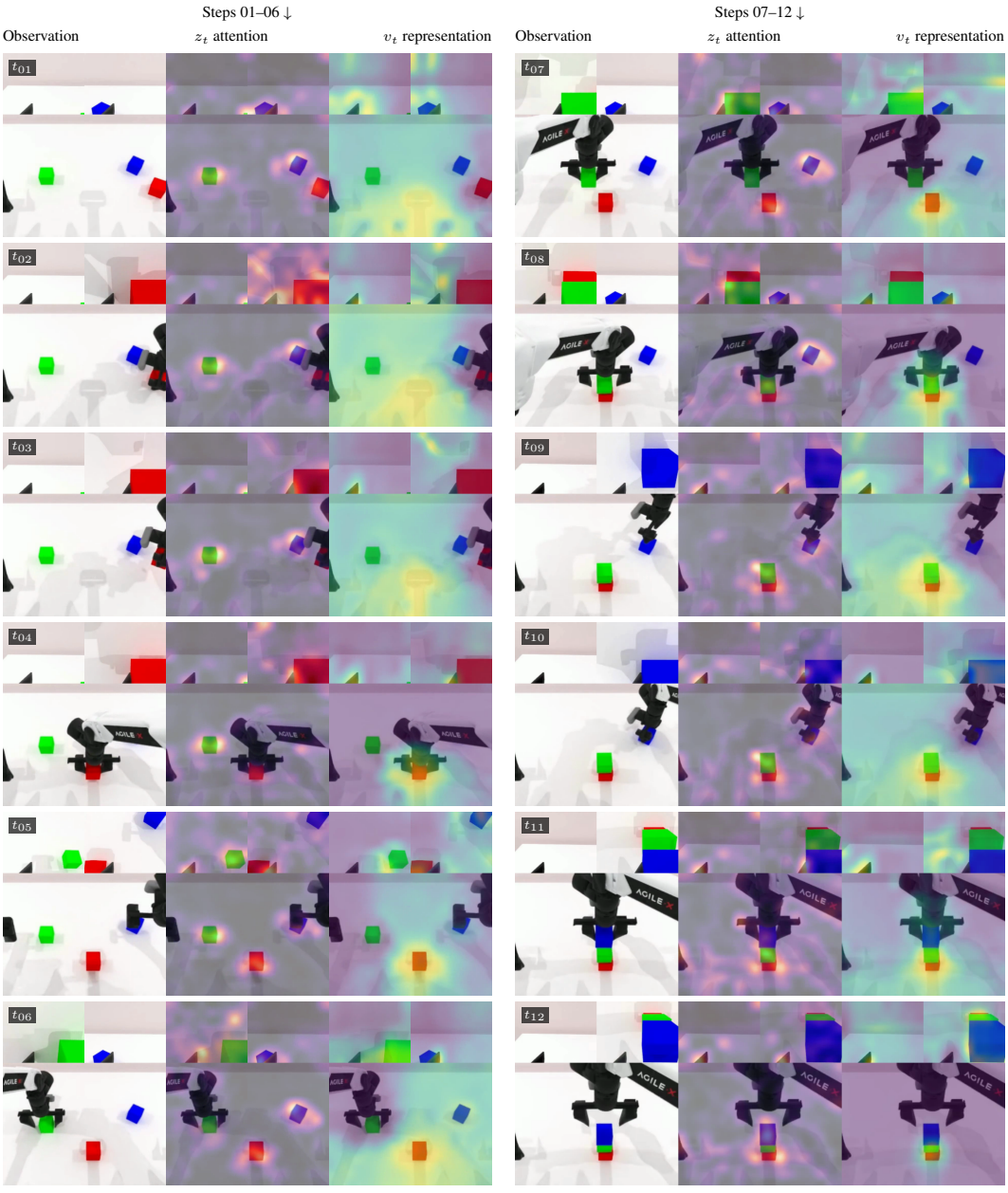


Figure A5: **RoboTwin episode: stack three blocks.** Uniformly sampled observations with the corresponding z_t attention and v_t representation maps.

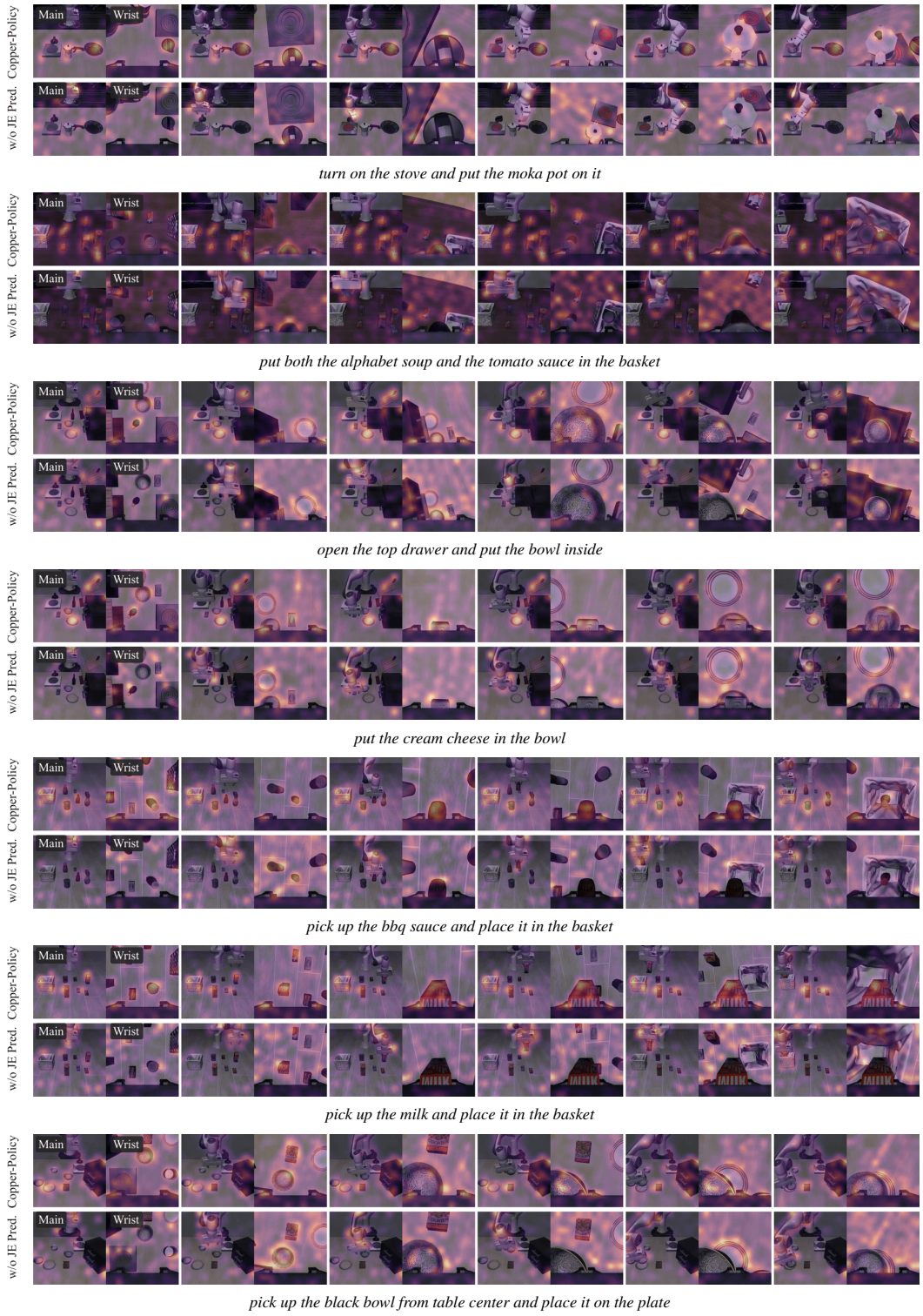


Figure A6: **Complete LIBERO z_t attention set.** Seven episodes spanning all four suites. Each block compares Copper-Policy (upper row) with *w/o Joint-Embedding Prediction* (lower row) on identical frames.

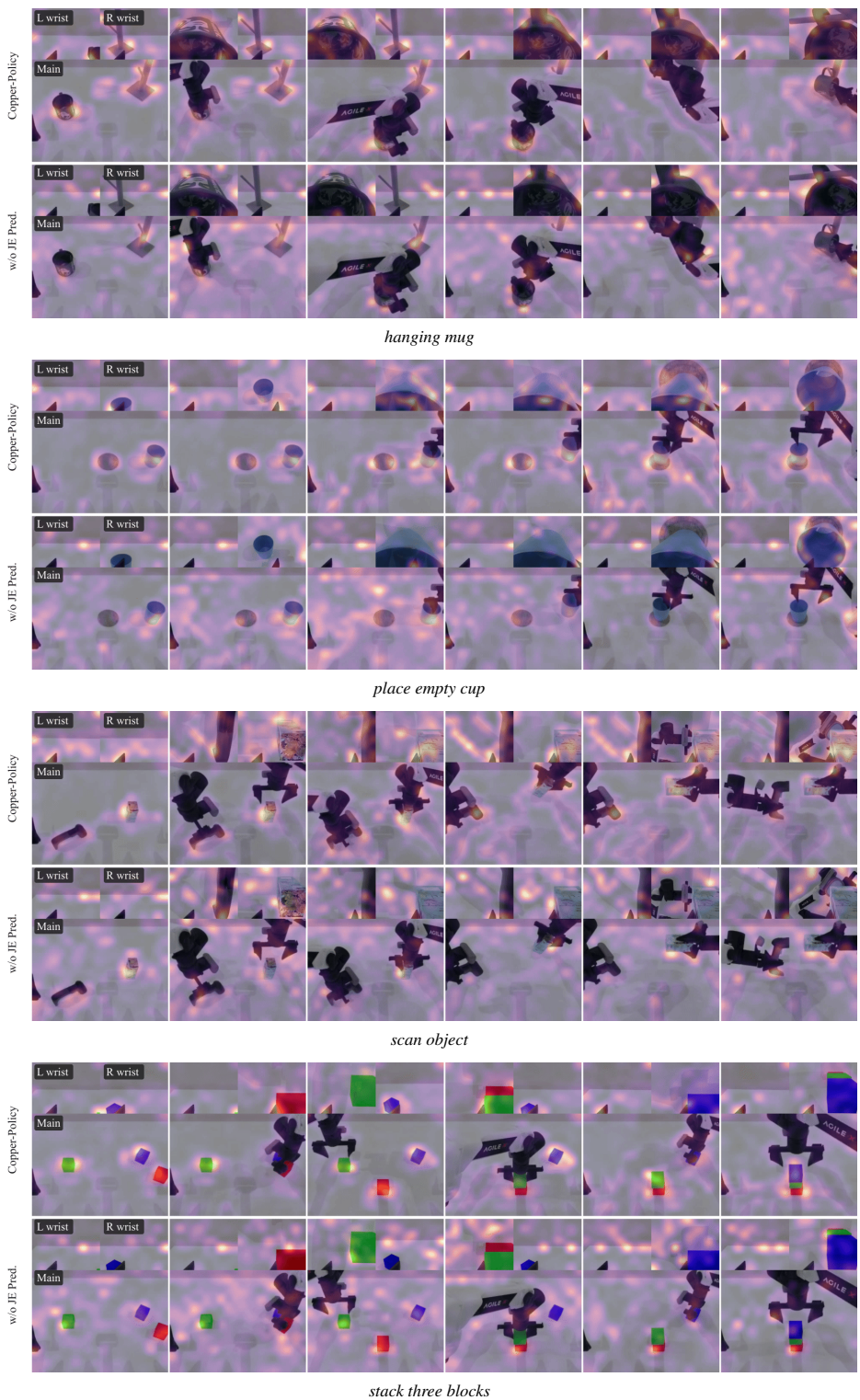


Figure A7: **Complete RoboTwin z_t attention set.** One episode from each of the four tasks used in the representation analysis. Each block compares Copper-Policy (upper row) with *w/o Joint-Embedding Prediction* (lower row) on identical frames; the T-shaped canvas places two wrist views above the front view.

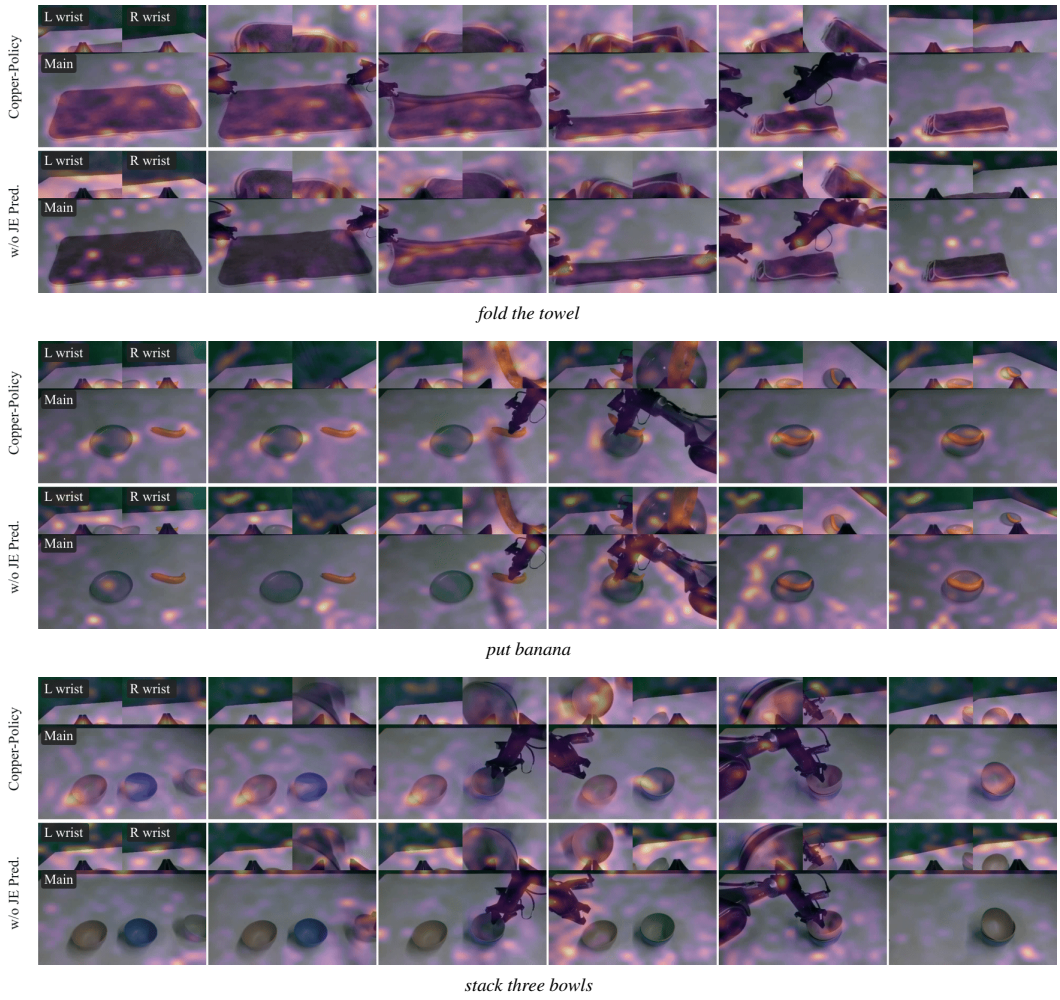


Figure A8: **Complete real-robot z_t attention set.** One episode from each of the three tasks, comparing Copper-Policy (upper row) with *w/o Joint-Embedding Prediction* (lower row) on identical frames.