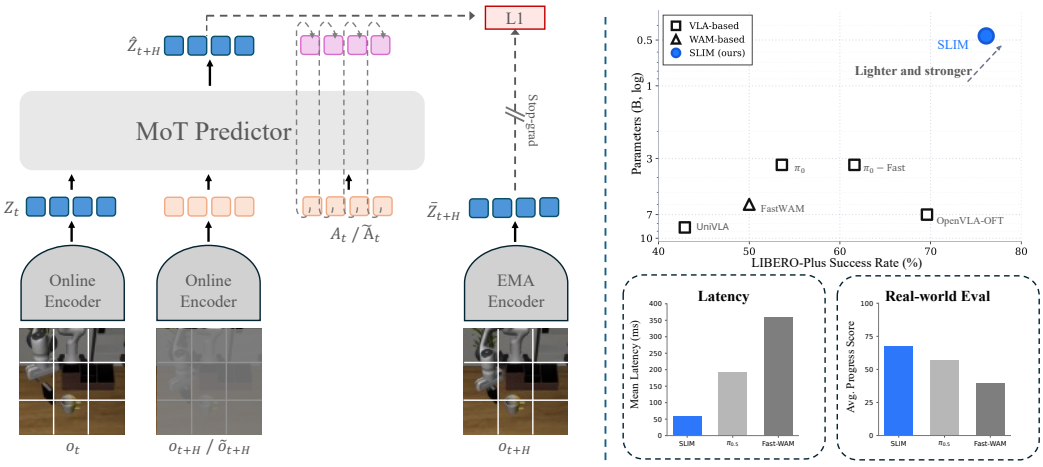
SLIM-0.5B: LEARNING ACTION-GROUNDED PREDICTIVE LATENTS FOR ROBOT MANIPULATION

Anonymous authors
Paper under double-blind review



(a) Overall Pipeline of Self-supervised Latent Interaction Model (b) Quantitative Results

Figure 1: Overview of SLIM. A compact MoT backbone models interactions among observation latents, continuous action tokens, and predictive future slots. Complementary inverse- and forward-dynamics objectives learn action-grounded predictive latents from robot trajectories, enabling robust, low-latency control with a compact policy.

ABSTRACT

Compact robot manipulation policies need representations that capture how actions change observations. Large multimodal backbones and pixel-level world models provide semantic and predictive structure, but can devote substantial computation to details beyond those needed for control. We propose **SLIM** (**S**elf-supervised **L**atent **I**nteraction **M**odel), a compact policy that learns action-grounded predictive representations within the same backbone used for action generation. A Mixture-of-Transformers (MoT) jointly models observation latents and continuous action tokens. In the first stage, SLIM learns these representations through self-supervised masked trajectory prediction. It reconstructs actions from current and future observations and predicts future visual representations from current observations and actions, using an exponential moving average (EMA) target encoder to provide the prediction targets. The second stage builds on the visual representations learned in the first stage and trains the shared backbone for action generation with flow matching, requiring neither future observations nor explicit future generation at deployment. With 0.47B trainable policy parameters, SLIM achieves competitive performance on LIBERO and CALVIN and reaches 77.45% success on zero-shot LIBERO-Plus. It also outperforms $\pi_{0.5}$ and Fast-WAM in average task progress across five real-world tasks and four evaluation settings. In our H100 inference benchmark, SLIM achieves $3.19\times$ and $5.95\times$ speedups per inference call over $\pi_{0.5}$ and Fast-WAM, respectively, while reducing per-call FLOPs by over 76% relative to both baselines. These results support bidirectional latent prediction as an effective training approach for compact manipulation policies.

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.

1 INTRODUCTION

Language-conditioned robot manipulation is increasingly shaped by two modeling paradigms. Vision-language-action (VLA) policies use large multimodal backbones for perception, instruction understanding, and action generation (Zitkovich et al., 2023; Driess et al., 2023; Kim et al., 2024; Black et al., 2024; Physical Intelligence et al., 2025; NVIDIA et al., 2025). These models bring semantic priors and open-domain language understanding, but typically learn action–observation relations implicitly through action supervision.

The second paradigm uses world models and world-action models to introduce predictive structure through future visual prediction or joint video–action modeling (Wu et al., 2024; Zhu et al., 2025; Ye et al., 2026; Ma et al., 2026). These approaches connect policy learning with future visual prediction, but pixel- or video-level prediction also models appearance details that may be irrelevant to control, such as background textures and lighting. Future generation adds computation and latency when performed during deployment.

Recent approaches learn predictive models in representation spaces (Chen et al., 2026; Wang et al., 2026a) or use future representations to supervise policies (Liu et al., 2026; Lyu et al., 2026; Zhang et al., 2025). We study a specific design within this direction: learning both forward and inverse action–observation relations inside the compact backbone that will later generate actions. We call such a representation *action-grounded*: it should support both inferring which action explains an observed transition and predicting the latent transition induced by an action.

We instantiate this design as **SLIM**, a **S**elf-supervised **L**atent **I**nteraction **M**odel. Conditioned on language instructions, a compact Mixture-of-Transformers (MoT) backbone (Liang et al., 2024) models interactions between observation latents and continuous action tokens. SLIM learns action-conditioned transitions in the same observation-latent space used for control, making dynamics an explicit training objective without a pixel decoder or a separate world-model state space.

In the first training stage, SLIM learns action-grounded representations through two complementary masked trajectory objectives: action reconstruction conditioned on current and future observations, and future-latent prediction conditioned on current observations and actions. Future-latent targets are produced by an exponential moving average (EMA) of the trainable online visual encoder. SLIM then builds on these learned representations and fine-tunes the same MoT backbone for action generation with flow matching (Chi et al., 2025; Liu et al., 2025; Black et al., 2024).

We evaluate SLIM on LIBERO, zero-shot LIBERO-Plus, CALVIN ABC→D, and real-world manipulation (Liu et al., 2023; Fei et al., 2025; Mees et al., 2022). SLIM achieves 77.45% success on zero-shot LIBERO-Plus alongside competitive performance on LIBERO and CALVIN. Averaged across five real-world tasks and four evaluation settings, it achieves higher task progress than $\pi_{0.5}$ and Fast-WAM. Under the controlled inference benchmark, SLIM also uses less GPU memory and achieves lower latency than these two baselines. These results support action-grounded latent prediction as an effective training approach for compact manipulation policies.

Our contributions are threefold. **First**, we develop SLIM, a compact policy with 0.47B trainable parameters that uses a shared MoT backbone for predictive representation learning and action generation. **Second**, we introduce self-supervised masked trajectory prediction that combines action reconstruction with future visual representation prediction against EMA-encoder targets, then transfers the learned representations to flow-matching control. **Third**, we demonstrate competitive performance across simulation and real-world manipulation, including zero-shot robustness and long-horizon composition, together with lower per-call latency and FLOPs.

2 RELATED WORK

Vision-language-action policies. Vision-language-action (VLA) and generalist robot policies combine visual perception, language understanding, and action generation in a shared model. Large-scale robot transformers established this paradigm across diverse demonstrations and embodiments (Brohan et al., 2022; Zitkovich et al., 2023; Driess et al., 2023; Open X-Embodiment Collaboration et al., 2023), followed by open-source and increasingly general VLA systems (Octo Model Team et al., 2024; Kim et al., 2024; Black et al., 2024; Physical Intelligence et al., 2025; NVIDIA et al., 2025). Although VLAs benefit from broad visual-language knowledge, they typically retain a large

VLM-style backbone at the center of every control step. Continuous robot actions are commonly modeled with diffusion or flow matching (Chi et al., 2025; Liu et al., 2025); recent VLAs adopt the same action-generation recipe while keeping the multimodal backbone for control. SLIM also adopts flow matching for action generation, but places the main control computation in a compact backbone that models interactions between observation latents and action tokens.

World action models and future-oriented policy representations. World action models connect actions with their consequences by predicting future observations or joint video-action trajectories (Wu et al., 2024; Zhu et al., 2025; Ye et al., 2026). Pixel-level prediction can model appearance details irrelevant to control, while test-time future generation adds control-loop latency. Recent work reduces this cost by removing test-time video prediction, compressing visual bandwidth, or replacing pixel futures with latent subgoals (Yuan et al., 2026; Tang et al., 2026; Chen et al., 2026). Closest to our setting, latent WAMs argue that compact latent targets are more useful for control than reconstruction-oriented futures (Chen et al., 2026; Nilaksh et al., 2026; Wang et al., 2026a). A parallel line supervises policy representations with future targets from frozen encoders or perception teachers (Liu et al., 2026; Lyu et al., 2026; Zhang et al., 2025; Sun et al., 2026b). SLIM learns bidirectional action–observation relations within the control backbone, using adaptive EMA targets rather than fixed external representations. Deployment requires neither a separate world-model rollout nor explicit future-latent conditioning.

JEPA and action-grounded predictive representations. Joint-embedding predictive architectures (JEPA) learn representations by predicting masked targets in embedding space rather than reconstructing pixels (LeCun, 2022; Assran et al., 2023), and video extensions apply this principle to temporal understanding and prediction (Bardes et al., 2024; Assran et al., 2025; Mur-Labadia et al., 2026). This view is closely related to actionable representation learning, which seeks to preserve factors relevant to decisions and control (Ghosh et al., 2019). Masked Trajectory Models (MTM) use masked trajectory reconstruction to support forward dynamics, inverse dynamics, and control within a single model (Wu et al., 2023). In robotics, VLA-JEPA adds JEPA-style future-latent prediction on a large VLM backbone (Sun et al., 2026a), while DeFI learns forward and inverse dynamics with separate modules (Zhang et al., 2026). SLIM combines bidirectional masked prediction with adaptive EMA targets in a compact MoT, then transfers the shared backbone and learned future-slot embeddings to flow-matching policy training.

3 METHOD

SLIM learns language-conditioned manipulation as action-grounded interaction in observation-latent space. The method separates three roles that are often entangled in large VLA or video-generative policies: language specifies the task, visual observations define the control state, and the action chunk specifies the continuous control trajectory to execute. Rather than using a large VLM as the main action backbone or reconstructing future pixels, SLIM first learns how actions and observation latents explain one another, then trains a compact flow-matching policy for action generation.

3.1 PROBLEM FORMULATION

Let y denote the language instruction and let $\ell = g_{\text{lang}}(y)$ be its encoded language condition, instantiated with T5-small (Raffel et al., 2020). At time t, the robot receives a multi-view visual observation $\mathbf{o}_t$ and proprioceptive state $\mathbf{q}_t$. The visual encoder maps the observation to a sequence of N_z observation-latent tokens

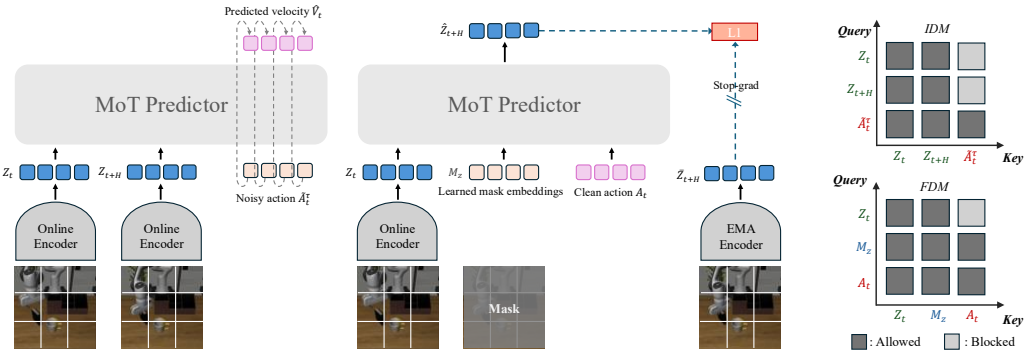
$$\mathbf{Z}_t = g_\psi(\mathbf{o}_t) \in \mathbb{R}^{N_z \times d_z}. \tag{1}$$

Here, d_z is the observation-latent feature dimension. In our implementation, g_ψ is initialized from DINOv2 (Oquab et al., 2024). The policy predicts an action chunk

$$\mathbf{A}_t = (\mathbf{a}_t, \ldots, \mathbf{a}_{t+H-1}), \tag{2}$$

where H is the action horizon. We use *action tokens* to denote the continuous embeddings of the action chunk; no action discretization is applied. The corresponding online future latent and its stable EMA target are

$$\mathbf{Z}_{t+H} = g_\psi(\mathbf{o}_{t+H}), \qquad \bar{\mathbf{Z}}_{t+H} = g_{\bar{\psi}}(\mathbf{o}_{t+H}), \tag{3}$$



(a) Inverse-Dynamics Objective (b) Forward-Dynamics Objective (c) Attention Masks

Figure 2: Action-grounded masked trajectory prediction. (a) Inverse dynamics predicts the flow velocity of noised actions conditioned on current and future observations. (b) Forward dynamics predicts masked future observation latents from the current observation and clean actions, supervised by a stop-gradient EMA target. (c) Objective-specific attention masks, with queries in rows and keys in columns. In FDM, future-slot queries M_z attend to current-observation and clean-action tokens. Auxiliary tokens are omitted for clarity.

where $g_{\bar{\psi}}$ is an exponential moving average (EMA) copy of g_ψ. The current and future online latents $\mathbf{Z}_t$ and $\mathbf{Z}_{t+H}$ therefore live in the same observation-latent space and differ only in temporal role.

3.2 Latent Interaction Policy

We instantiate the policy backbone as a compact Mixture-of-Transformers (MoT) (Liang et al., 2024). As illustrated in Figure 3, MoT maintains two interacting streams: an observation stream and an action stream. Before entering the MoT trunk, the inputs are mapped to its shared hidden dimension d ($d = 768$ in our implementation). Each observation-latent token is linearly projected to d, and the proprioceptive state $\mathbf{q}_t$ is mapped by a separate linear projection to a single state token that is prepended to the observation stream. For the action stream, a three-layer MLP encoder first projects each continuous action vector to d, concatenates it with a sinusoidal embedding of the flow timestep, and then maps the result to an action token in $\mathbb{R}^d$. The observation stream is therefore grounded in both the current observation latent $\mathbf{Z}_t$ and proprioceptive state $\mathbf{q}_t$. The observation stream also maintains N_z learned future-slot embeddings, collected as $\mathbf{M}_z = [\mathbf{m}_z^{(1)}, \ldots, \mathbf{m}_z^{(N_z)}]^\top \in \mathbb{R}^{N_z \times d}$. They are shared across examples but distinct across future-token positions. The future positions are filled with the projected clean future latent $\mathbf{Z}_{t+H}$ in the action-masked branch and with $\mathbf{M}_z$ in the future-latent-masked and policy branches. The action stream carries the action variable used by the objective: the noised action chunk $\tilde{\mathbf{A}}_t^\tau$ for flow-matching action prediction, or the clean action chunk $\mathbf{A}_t$ for future-observation-latent prediction. Language ℓ is injected through per-stream cross-attention rather than being treated as part of the joint observation-action stream. We denote the action-velocity predictor by v_θ and the future-observation-latent predictor by p_θ; both use the same MoT backbone with objective-specific inputs.

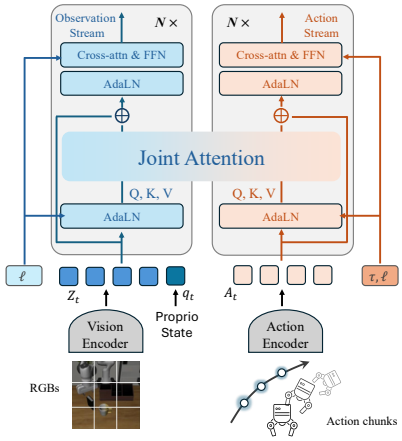


Figure 3: MoT backbone with observation-action joint attention and per-stream language cross-attention.

Within each MoT block, the two streams produce stream-specific queries, keys, and values, and then interact through a shared joint-attention operation. The attention output is split back into observation-side and action-side tokens, followed by per-stream language cross-attention and feed-

forward layers. This design makes language a task condition while reserving the main computation for observation-action interaction. During flow sampling, the current observation latents, proprioception, and language remain fixed as input conditions. At each sampling step, the MoT processes these conditions together with the current action iterate and flow timestep to predict the action velocity.

3.3 Action-Grounded Training

SLIM uses two training stages with a shared observation–action interaction backbone. Stage 1 learns predictive representations through complementary inverse- and forward-dynamics objectives. The inverse-dynamics branch receives flow-noised actions and encoded future observations, whereas the forward-dynamics branch receives clean actions and learned future-slot embeddings, with future observations supplying the EMA targets. Stage 2 trains the flow-matching policy using learned future-slot embeddings in place of encoded future observations. Current observation latents, proprioception, and language condition both stages.

Stage-1 masked trajectory prediction. Stage 1 learns predictive observation latents by masking trajectory variables and reconstructing them from the remaining context, following the representation-space prediction principle of JEPA-style objectives (Assran et al., 2023; Bardes et al., 2024). The supervision comes from robot trajectories themselves, without additional annotations or pixel reconstruction. We use two complementary masked-prediction branches.

The inverse-dynamics branch reconstructs actions through conditional flow matching (Lipman et al., 2023), conditioned on the current observation latent $\mathbf{Z}_t$, the clean future observation latent $\mathbf{Z}_{t+H}$, proprioception $\mathbf{q}_t$, and language ℓ. Action masking uses Gaussian noise and a transformed Beta flow-time distribution p_τ (Appendix A.6):

$$\epsilon \sim \mathcal{N}(0, I), \qquad \tau \sim p_\tau, \qquad \tilde{\mathbf{A}}_t^\tau = (1-\tau)\epsilon + \tau \mathbf{A}_t, \qquad \mathbf{V}_t^\star = \mathbf{A}_t - \epsilon. \tag{4}$$

The inverse-dynamics-style objective is

$$\mathcal{L}_{\mathrm{IDM}} = \mathbb{E}\left[\left\| v_\theta(\tilde{\mathbf{A}}_t^\tau, \tau \mid \mathbf{Z}_t, \mathbf{Z}_{t+H}, \mathbf{q}_t, \ell) - \mathbf{V}_t^\star \right\|_2^2\right]. \tag{5}$$

This branch asks which action chunk explains the transition from $\mathbf{Z}_t$ to $\mathbf{Z}_{t+H}$.

When the future observation latent is masked, SLIM replaces its token positions with $\mathbf{M}_z$ and conditions on the current observation latent and clean action chunk. The model predicts

$$\hat{\mathbf{Z}}_{t+H} = p_\theta(\mathbf{M}_z \mid \mathbf{Z}_t, \mathbf{A}_t, \mathbf{q}_t, \ell), \tag{6}$$

and aligns it to the detached EMA target:

$$\mathcal{L}_{\mathrm{FDM}} = \left\| \mathrm{LN}(\hat{\mathbf{Z}}_{t+H}) - \mathrm{LN}(\mathrm{sg}(\bar{\mathbf{Z}}_{t+H})) \right\|_1. \tag{7}$$

Here, LN denotes per-token layer normalization over the feature dimension without learnable affine parameters, and sg denotes stop-gradient. This objective trains the model to predict future observation representations from the current observation and action chunk. The full Stage-1 loss is

$$\mathcal{L}_{\mathrm{stage1}} = \lambda_{\mathrm{IDM}} \mathcal{L}_{\mathrm{IDM}} + \lambda_{\mathrm{FDM}} \mathcal{L}_{\mathrm{FDM}}. \tag{8}$$

Together, the two branches make the learned latent space action-grounded: actions must explain future observation latents, and future observation latents must constrain action reconstruction. These inverse- and forward-dynamics-style signals are related to dynamics pretraining objectives for robot learning (Zhang et al., 2026).

Stage-2 flow-matching policy training. After masked trajectory prediction, SLIM trains the final policy for action generation. This training setup matches inference: the policy conditions on the current observation latent $\mathbf{Z}_t$, learned future-slot embeddings $\mathbf{M}_z$, proprioception $\mathbf{q}_t$, language ℓ, and the noised action chunk used by flow sampling. No future observation latent is provided to the policy at inference time. Using the same flow interpolation as in Eq. 4, the Stage-2 objective is

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}\left[\left\| v_\theta(\tilde{\mathbf{A}}_t^\tau, \tau \mid \mathbf{Z}_t, \mathbf{M}_z, \mathbf{q}_t, \ell) - \mathbf{V}_t^\star \right\|_2^2\right]. \tag{9}$$

The training objective in Stage 2 is simply

$$\mathcal{L}_{\text{stage2}} = \mathcal{L}_{\text{FM}}. \tag{10}$$

Stage 2 retains the online visual encoder, MoT backbone, and learned future-slot embeddings from Stage 1, transferring the predictive representation and interaction structure to action generation. These components are fine-tuned with the flow-matching objective, without future-observation inputs or future-latent supervision. During inference, the action stream attends to the observation-side hidden states, including those computed from the learned future slots, without a separate future-generation rollout or latent decoding step.

4 EXPERIMENTS

We evaluate SLIM in simulation and on real-world tasks. The simulation study tests nominal language-conditioned manipulation, robustness under controlled distribution shifts, and long-horizon composition in an unseen environment. The real-world study is designed to test whether the same compact latent interaction policy can be trained and executed on physical manipulation tasks. Figure 4 summarizes the evaluation settings.

4.1 SIMULATION EVALUATION

Benchmarks. We use three complementary simulation benchmarks. LIBERO evaluates language-conditioned manipulation on four standard task suites: LIBERO-10, LIBERO-Object, LIBERO-Spatial, and LIBERO-Goal (Liu et al., 2023). LIBERO-Plus builds controlled perturbations on top of LIBERO, including camera, robot initial state, language, lighting, background, sensor noise, and object-layout shifts (Fei et al., 2025). CALVIN ABC→D evaluates long-horizon language-conditioned task composition: the policy is trained on environments A/B/C and evaluated in the unseen environment D, where each evaluation chain contains up to five sequential language instructions (Mees et al., 2022). We report success rate for LIBERO and LIBERO-Plus, and average sequence length for CALVIN.

Baselines. We compare against representative vision-language-action (VLA) policies and WAM-style methods. Tables report model size and P.T., denoting additional policy or world-model pre-training on robot data beyond the target benchmark; generic vision-language or video backbone initialization is excluded. The LIBERO-family comparisons include OpenVLA-OFT (Kim et al., 2025), NORA (Hung et al., 2025), WorldVLA (Cen et al., 2025), UnifiedVLA (Wang et al., 2026b), and RIPT-VLA (Tan et al., 2026). The CALVIN comparisons further include RoboFlamingo (Li et al., 2024), RoboDual (Bu et al., 2024), FLOWER (Reuss et al., 2025), Seer-Large (Tian et al., 2025), GR-MG (Li et al., 2025), UniPi (Du et al., 2023), and VPP (Hu et al., 2025). For readability, the main tables report a compact baseline subset; detailed original-LIBERO results are provided in Appendix Table 3.

Training and testing setting. For the LIBERO experiments, Stage 1 uses LIBERO-90 together with the four target suites, while Stage 2 uses only LIBERO-10, LIBERO-Object, LIBERO-Spatial, and LIBERO-Goal. The resulting checkpoint is evaluated on LIBERO-Plus without adaptation or training on its perturbation cases. For CALVIN, SLIM is trained on the ABC training environments and evaluated on the held-out D environment.

Results. Tables 1 and 2 summarize the simulation results, with detailed original-LIBERO results provided in Appendix Table 3. SLIM achieves **97.5%** success on original LIBERO, remaining competitive with the strongest reported methods. On zero-shot LIBERO-Plus, it achieves **77.45%**, exceeding OpenVLA-OFT and Fast-WAM by 7.85 and 27.45 percentage points, respectively. It ranks first among the listed methods on camera, language, and sensor-noise perturbations, while VLA-JEPA achieves the highest overall score. Robot-initial-state perturbations remain challenging for SLIM, with 36.90% success compared with 67.1% for VLA-JEPA.

Table 2 reports CALVIN ABC→D. Unlike LIBERO, this benchmark uses average sequence length rather than success rate. SLIM achieves 4.556 out of 5 on the unseen-environment split, suggesting that the learned latent interaction policy supports long-horizon language-conditioned task composition beyond the LIBERO family.

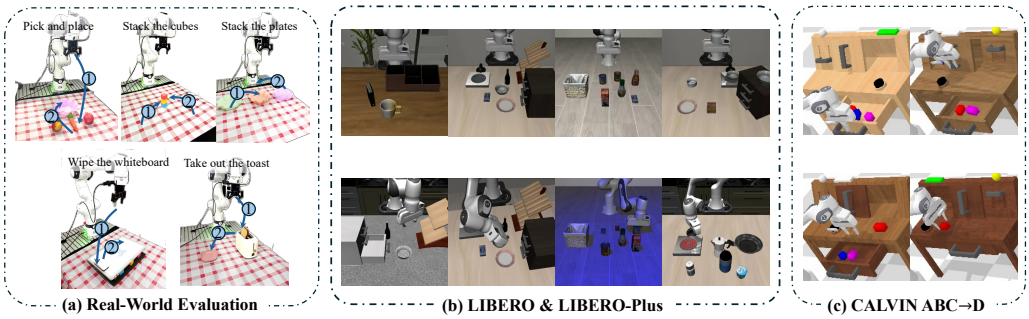


Figure 4: Overview of the evaluation settings. (a) Representative physical manipulation tasks used in the real-world evaluation. (b) LIBERO and LIBERO-Plus evaluate language-conditioned manipulation and robustness under controlled perturbations. (c) CALVIN ABC→D evaluates long-horizon task composition in an unseen environment.

Table 1: Results on LIBERO and LIBERO-Plus (%). LIBERO reports the original-benchmark overall score, while LIBERO-Plus evaluates zero-shot robustness under seven perturbations. VLA-JEPA scores follow Sun et al. (2026a). P.T. denotes policy/world-model pretraining on robot data outside the target benchmark, excluding generic vision-language or video initialization.

Method	P.T.	Size	LIBERO	Camera	Robot	Language	Light	Background	Noise	Layout	Overall
VLA baselines											
OpenVLA	✓	7B	76.5	0.8	3.5	23.0	8.1	34.8	15.2	28.5	15.6
OpenVLA-OFT	✓	7B	97.1	56.4	31.9	79.5	88.7	**93.3**	75.8	74.3	69.6
π_0	✓	3.3B	94.1	61.0	40.8	63.7	89.3	84.1	**80.1**	75.9	69.3
NORA	✓	3B	87.9	2.2	37.0	65.1	45.7	58.6	12.8	62.1	39.0
WorldVLA	×	7B	79.1	0.1	27.9	41.6	43.7	17.1	11.0	38.0	25.0
UnifiedVLA	✓	8.5B	95.5	1.8	**46.2**	69.5	69.0	81.0	21.2	31.9	42.9
RIPT-VLA	✓	7B	**97.5**	55.2	31.2	77.6	88.4	91.6	73.5	74.2	68.4
WAM baselines											
Fast-WAM	×	6B	**97.6**	16.4	44.5	68.9	78.2	53.7	37.7	60.7	50.0
VLA-JEPA	✓	3B	97.2	**63.3**	**67.1**	**85.4**	**95.6**	**93.6**	66.3	**85.1**	**79.5**
Ours											
SLIM	×	**0.47B**	**97.5**	**70.73**	36.90	**87.57**	**94.75**	92.01	**86.07**	**83.21**	**77.45**

4.2 Real-World Evaluation

We further evaluate SLIM on a real-world manipulation suite. The goal of this study is to test whether the compact latent interaction policy remains effective when trained from physical demonstrations and evaluated under visual and scene-level disturbances. We compare SLIM with two representative baselines, $\pi_{0.5}$ and Fast-WAM, which respectively represent a strong VLA policy and a WAM-style policy.

Data and training. We design five real-world manipulation tasks: placing a carrot into a bowl, stacking three plates, taking a piece of toast out of a toaster and placing it onto a plate, stacking blocks, and wiping a whiteboard. For each task, we collect 150 demonstrations, resulting in 750 demonstrations in total. All tasks are mixed during training, so each policy is trained as a multi-task real-world policy rather than as five separately trained task-specific policies. SLIM follows the same two-stage recipe as in simulation: Stage-1 masked trajectory learning is applied to the collected robot trajectories, followed by flow-matching policy training for action generation. Both baselines are trained on the same 750-demonstration mixture. For $\pi_{0.5}$, we initialize from the official DROID-fine-tuned checkpoint (`pi05_droid`) and further fine-tune it on our demonstrations.

Table 2: Results on CALVIN ABC→D. Task 1–5 report sequential success rates (%), and Avg. Length is the primary metric. P.T. denotes policy/world-model pretraining on robot data outside the target benchmark, excluding generic vision-language or video initialization. Model sizes are reported in billions of parameters (B); "–" indicates unavailable data.

Method	P.T.	Size	Task 1	Task 2	Task 3	Task 4	Task 5	Avg. Length
VLA baselines								
RoboFlamingo	×	3B	82.4	61.9	46.6	33.1	23.5	2.47
OpenVLA	✓	7B	91.3	77.8	62.0	52.1	43.5	3.27
RoboDual	✓	7.02B	94.4	82.7	72.1	62.4	54.4	3.66
UnifiedVLA	✓	8.5B	98.9	94.8	89.0	82.8	75.1	4.41
FLOWER	✓	0.95B	**99.4**	**95.8**	**90.7**	**84.9**	77.8	**4.53**
Seer-Large	✓	0.57B	96.3	91.6	86.1	80.3	74.0	4.28
GR-MG	×	–	96.8	89.3	81.5	72.7	64.4	4.04
WAM baselines								
GR-1	×	0.20B	85.4	71.2	59.6	49.7	40.1	3.06
UniPi	×	–	56.0	16.0	8.0	8.0	4.0	0.92
VPP	✓	1.5B	96.5	90.9	86.6	82.0	76.9	4.33
DreamVLA	✓	–	98.2	94.6	89.5	83.4	**78.1**	4.44
Ours								
SLIM	×	**0.47B**	**99.3**	**96.7**	**92.3**	**87.1**	**80.2**	**4.556**

Evaluation protocol. We evaluate each policy on the five real-world tasks under the nominal setting and three perturbation dimensions: background texture changes, lighting changes, and added distractor objects. Each task-condition pair is evaluated with 10 trials. Complete task-specific OOD configurations are provided in Appendix Figure 8.

Results. For each task-condition pair, the progress score is 100 times the mean trial score over 10 trials, using task-specific milestones defined in Appendix A.8. Averaged across all five tasks and four settings, SLIM achieves 67.8, compared with 56.8 for $\pi_{0.5}$ and 40.0 for Fast-WAM (Figure 5). SLIM obtains the highest average progress under nominal, distractor, and lighting conditions. Under background changes, it scores 49, below $\pi_{0.5}$ at 54 and above Fast-WAM at 2. Appendix Figure 10 provides the complete task-wise results.

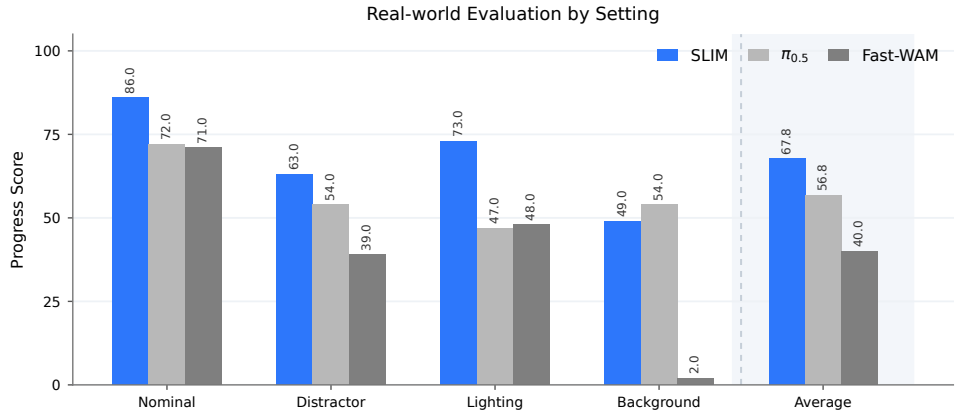


Figure 5: Average real-world progress by evaluation setting. Each setting averages the five tasks, and the final group averages all four settings. Task-wise results and scoring milestones are provided in Appendix A.8.

4.3 Ablation Study

We examine the contribution of Stage-1 training, the balance between inverse and forward objectives, and the prediction-target design. The Stage-2-only baseline is trained for exactly the same total number of optimization steps as Stage 1 and Stage 2 combined. Figure 6 shows that the full two-stage recipe outperforms Stage-2-only training on LIBERO-Plus and CALVIN. Among the tested loss ratios, an IDM:FDM weight of $0.125{:}1$ provides the strongest joint result, improving on the FDM-only $(0{:}1)$ variant. The EMA target also improves LIBERO-Plus success from 66.82% to 77.45% and CALVIN average sequence length from 4.382 to 4.556 relative to the no-EMA variant, which uses stop-gradient targets from the online encoder.

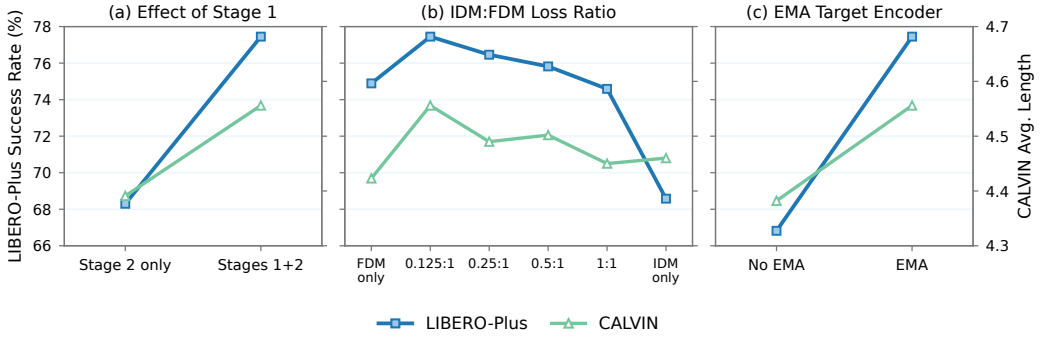


Figure 6: Ablations of the action-grounded masked-prediction pipeline. We compare Stage-2-only training against the full two-stage recipe, vary the IDM:FDM loss ratio, and evaluate the EMA target encoder on LIBERO-Plus and CALVIN.

Keeping the target encoder fixed at its initial DINOv2 weights reduces LIBERO-Plus success from 77.45% to 71.81%, while the online encoder remains trainable in both variants (Appendix A.2). This comparison supports updating the prediction target during representation learning. Separately, removing EMA reduces the batch-level effective rank of future latents from 61.28 to 13.95 despite lowering normalized L1 prediction error (Appendix A.3), indicating that lower prediction error alone does not imply better representations.

4.4 Analysis

Action-to-observation attention. In the illustrated sequences, SLIM's action-to-observation attention more consistently follows task-relevant objects, the gripper, and their contact regions than the policy trained without Stage 1 (Appendix A.4). These qualitative patterns complement the Stage-1 ablation and suggest a shift toward manipulation-relevant visual structure.

Latency, memory, and computation. In our PyTorch eager policy-inference benchmark on an NVIDIA H100 80GB, SLIM averages 60.6 ms per call, uses 4.26 GiB of peak GPU memory, and requires 490.73 GFLOPs per action chunk. Compared with $\pi_{0.5}$ and Fast-WAM, respectively, it is $3.19\times$ and $5.95\times$ faster, uses $1.86\times$ and $3.20\times$ less memory, and requires $9.61\times$ and $4.26\times$ fewer FLOPs; Appendix A.7 provides the complete benchmark protocol and model-specific settings.

5 Conclusion

We introduced SLIM, a compact policy that learns action-grounded predictive representations through complementary inverse-dynamics and latent forward-prediction objectives. The learned visual representations and shared MoT backbone are transferred to flow-matching control, without requiring future observations or explicit future generation at deployment. Across LIBERO, zero-shot LIBERO-Plus, CALVIN ABC→D, and real-world manipulation, SLIM matches or exceeds representative VLA and WAM baselines with fewer parameters, lower latency, and lower memory. Our evaluation covers one model scale; future work should examine scaling across model capacity, training data, and embodiment diversity.

AI USE STATEMENT

We used generative AI tools to assist with drafting and revising manuscript text, discussing and interpreting experimental results, and preparing the anonymous code artifact and its documentation. Generative AI was not used to generate experimental data or numerical results. The authors take responsibility for the final manuscript, scientific claims, code, and reported results.

ETHICS STATEMENT

This work studies robot manipulation using public simulation benchmarks and real-world robot demonstrations. The real-world demonstrations were collected by members of the research team, without recruiting external participants or collecting personally identifiable information. Physical robot experiments were conducted under human supervision with emergency-stop access. Learned manipulation policies can produce unsafe actions under unfamiliar conditions; deployment beyond the evaluated settings requires application-specific safety assessment and safeguards.

REPRODUCIBILITY STATEMENT

Section 3 describes the model and training objectives, and Section 4 specifies the experimental setup. Appendix A.6 provides model inputs, optimization settings, and benchmark-specific protocols. Appendices A.7 and A.8 document the inference benchmark and real-world evaluation settings and scoring criteria. Our implementation, simulation training and evaluation scripts, configurations, and environment and dataset setup instructions are available at `https://anonymous.4open.science/r/SLIM-review-70F0/`. Selected successful real-world demonstrations are available on our anonymous project page. Trained SLIM checkpoints and the collected real-world demonstration dataset are not included in the review artifact.

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, 2023. URL `https://arxiv.org/abs/2301.08243`.

Mido Assran et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

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, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2024. URL `https://arxiv.org/abs/2410.24164`.

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

Qingwen Bu, Hongyang Li, Li Chen, Jisong Cai, Jia Zeng, Heming Cui, Maoqing Yao, and Yu Qiao. Towards synergistic, generalized, and efficient dual-system for robotic manipulation. *arXiv preprint arXiv:2410.08001*, 2024.

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

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

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

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. PaLM-E: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

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 *Advances in Neural Information Processing Systems*, volume 36, pp. 9156–9172, 2023.

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.

Dibya Ghosh, Abhishek Gupta, and Sergey Levine. Learning actionable representations with goal-conditioned policies, 2019. URL https://arxiv.org/abs/1811.07819.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 24328–24346. PMLR, 2025. URL https://proceedings.mlr.press/v267/hu25g.html.

Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U-Xuan Tan, Navonil Majumder, and Soujanya Poria. NORA: A small open-sourced generalist vision language action model for embodied tasks. *arXiv preprint arXiv:2504.19854*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag 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, 2024. URL https://arxiv.org/abs/2406.09246.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Robotics: Science and Systems*, 2025. URL https://roboticsconference.org/2025/program/papers/17/.

Yann LeCun. A path towards autonomous machine intelligence, 2022. URL https://openreview.net/pdf?id=BZ5a1r-kVsf. Version 0.9.2.

Peiyan Li, Hongtao Wu, Yan Huang, Chilam Cheang, Liang Wang, and Tao Kong. GR-MG: Leveraging partially-annotated data via multi-modal goal-conditioned policy. *IEEE Robotics and Automation Letters*, 10(2):1912–1919, 2025.

Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, Hang Li, and Tao Kong. Vision-language foundation models as effective robot imitators. In *International Conference on Learning Representations*, 2024.

Weixin Liang, Lili Yu, Liang Luo, Srini Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen-tau Yih, Luke Zettlemoyer, et al. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *Transactions on Machine Learning Research*, 2024.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

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

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: A diffusion foundation model for bimanual manipulation. In *International Conference on Learning Representations*, 2025.

Zhuoyang Liu, Jiaming Liu, Hao Chen, Jiale Yu, Ziyu Guo, Chengkai Hou, Chenyang Gu, Xiangju Mi, Renrui Zhang, Kun Wu, et al. LaST$_0$: Latent spatio-temporal chain-of-thought for robotic vision-language-action model. In *Forty-third International Conference on Machine Learning*, 2026.

Jiangran Lyu, Kai Liu, Xuheng Zhang, Haoran Liao, Yusen Feng, Wenxuan Zhu, Tingrui Shen, Jiayi Chen, Jiazhao Zhang, Yifei Dong, et al. LDA-1B: Scaling latent dynamics action model via universal embodied data ingestion. *arXiv preprint arXiv:2602.12215*, 2026.

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

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

Lorenzo Mur-Labadia, Matthew Muckley, Amir Bar, Mido Assran, Koustuv Sinha, Mike Rabbat, Yann LeCun, Nicolas Ballas, and Adrien Bardes. V-JEPA 2.1: Unlocking dense features in video self-supervised learning. *arXiv preprint arXiv:2603.14482*, 2026.

Nilaksh, Saurav Jha, Artem Zholus, and Sarath Chandar. Reconstruction or semantics? what makes a latent space useful for robotic world models. *arXiv preprint arXiv:2605.06388*, 2026.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Jim Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL `https://arxiv.org/abs/2503.14734`.

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

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, et al. Open x-embodiment: Robotic learning datasets and RT-X models. *arXiv preprint arXiv:2310.08864*, 2023.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

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.

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.

Moritz Reuss, Hongyi Zhou, Marcel Rühle, Ömer Erdinç Yağmurlu, Fabian Otto, and Rudolf Lioutikov. FLOWER: Democratizing generalist robot policies with efficient vision-language-action flow policies. *arXiv preprint arXiv:2509.04996*, 2025.

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

Zhenguo Sun, Yu Sun, Hande Huang, and Alois Knoll. ω-EVA: Envision, verify, and act with latent interactive world models, 2026b. URL https://arxiv.org/abs/2606.09457.

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models. In *International Conference on Learning Representations*, 2026.

Zuojin Tang, Shengchao Yuan, Xiaoxin Bai, Zhiyuan Jing, De Ma, Gang Pan, and Bin Liu. One token per frame: Reconsidering visual bandwidth in world models for vla policy, 2026. URL https://arxiv.org/abs/2605.07931.

Yang Tian, Sizhe Yang, Jia Zeng, Ping Wang, Dahua Lin, Hao Dong, and Jiangmiao Pang. Predictive inverse dynamics models are scalable learners for robotic manipulation. In *International Conference on Learning Representations*, 2025.

Junke Wang, Qihang Zhang, Shuai Yang, Yiming Luo, Yujun Shen, Zuxuan Wu, Yu-Gang Jiang, and Yinghao Xu. Repwam: World action modeling with representation visual-action tokenizers. *arXiv preprint arXiv:2606.13674*, 2026a.

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

Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *International Conference on Learning Representations*, 2024.

Philipp Wu, Arjun Majumdar, Kevin Stone, Yixin Lin, Igor Mordatch, Pieter Abbeel, and Aravind Rajeswaran. Masked trajectory models for prediction, representation, and control. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 37607–37623. PMLR, 2023. URL https://proceedings.mlr.press/v202/wu23o.html.

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

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.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, He Wang, Zhizheng Zhang, et al. DreamVLA: A vision-language-action model dreamed with comprehensive world knowledge. *Advances in Neural Information Processing Systems*, 38:24195–24228, 2025.

Wenyao Zhang, Bozhou Zhang, Zekun Qi, Wenjun Zeng, Xin Jin, and Li Zhang. Disentangled robot learning via separate forward and inverse dynamics pretraining, 2026. URL https://arxiv.org/abs/2604.16391.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets, 2025. URL https://arxiv.org/abs/2504.02792.

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

A Appendix

A.1 Detailed Original-LIBERO Results

Table 3 reports suite-level results on the original LIBERO benchmark. The main paper combines the original-benchmark overall score with the zero-shot LIBERO-Plus perturbation results for a compact robustness comparison.

Table 3: Original LIBERO baseline comparison. All numbers are success rates in percent. VLA-JEPA scores follow Sun et al. (2026a). P.T. denotes policy/world-model pretraining on robot data outside the target benchmark, excluding generic vision-language or video initialization.

Method	P.T.	Size	Long	Spatial	Object	Goal	Overall
VLA baselines							
OpenVLA	✓	7B	53.7	84.7	88.4	79.2	76.5
OpenVLA-OFT	✓	7B	94.5	97.6	98.4	97.9	97.1
π_0	✓	3.3B	85.2	96.8	98.8	95.8	94.1
$\pi_{0.5}$	✓	3.3B	92.4	98.8	98.2	**98.0**	96.9
NORA	✓	3B	74.6	92.2	95.4	89.4	87.9
WorldVLA	×	7B	59.0	85.6	89.0	82.6	79.1
UnifiedVLA	✓	8.5B	94.0	95.4	98.8	93.6	95.5
RIPT-VLA	✓	7B	93.8	**99.0**	98.6	**98.6**	**97.5**
WAM baselines							
Fast-WAM	×	6B	**95.2**	98.2	**100.0**	97.0	**97.6**
VLA-JEPA	✓	3B	**95.8**	96.2	**99.6**	97.2	97.2
Ours							
SLIM	×	**0.47B**	94.4	**99.4**	99.4	96.8	**97.5**

A.2 Fixed DINOv2 Target Ablation

To separate the benefit of a detached target branch from that of updating the target with the online representation, we rerun the canonical LIBERO recipe while keeping the Stage-1 target encoder fixed at its initial DINOv2-B/14 weights. The online visual encoder remains trainable; only the target update is disabled by setting the EMA momentum to 1.0. Stage 2 and the evaluation protocol are otherwise unchanged. As shown in Table 4, the fixed target reduces original-LIBERO success by 0.8 points and LIBERO-Plus success by 5.64 points. The larger robustness drop indicates that the EMA benefit is not explained solely by stop-gradient supervision: allowing the target to slowly track the task-adapted online encoder is important for transferable predictive features.

Table 4: Stage-1 visual-target ablation. Original LIBERO and LIBERO-Plus report success rates in percent. The online visual encoder is trainable in both variants.

Target variant	Target update	LIBERO	LIBERO-Plus
EMA target	$m = 0.999$	97.5	77.45
Fixed DINOv2 target	$m = 1.0$	96.7	71.81

A.3 EMA Target Representation Diagnostics

Table 5 further examines the EMA target through representation diagnostics. In the no-EMA variant, future-latent targets are computed by the trainable online visual encoder rather than an EMA target encoder. Stop-gradient is retained on these targets, so the prediction loss does not backpropagate through the target computation.

Diagnostic metrics. For future-target latents of shape $B \times Q \times D$, where B is the batch size, Q is the number of tokens per sample, and D is the feature dimension, we pool all BQ tokens and center

each feature across the pooled tokens. Let λ_i denote the eigenvalues of the resulting sample covariance matrix and $p_i = \lambda_i / \sum_j \lambda_j$. The batch-level effective rank is $\exp(-\sum_i p_i \log p_i)$. Top-1 and top-5 energy concentrations are the fractions of total covariance energy captured by the largest one and five eigenvalues, respectively. Token cosine similarity is averaged over all distinct token pairs within each sample and then across the batch. Future-latent prediction error is the mean absolute difference between predicted and target latents after independently applying non-affine LayerNorm along each token's feature dimension, averaged over samples, tokens, and features.

Results. Averaged over the final 20 recorded measurements during Stage 1, the no-EMA variant has a lower batch-level effective rank of future latents than the EMA variant (13.95 versus 61.28) and higher within-sample token cosine similarity (0.352 versus 0.071). Top-1 and top-5 energy concentrations also increase, from 0.097 to 0.395 and from 0.362 to 0.674, respectively. Despite a lower normalized L1 prediction error (0.166 versus 0.245), LIBERO-Plus success decreases from 77.45% to 66.82%. Together, these diagnostics associate EMA targets with greater latent diversity and better downstream robustness, while showing that lower prediction error alone does not imply better representations.

Table 5: EMA target representation diagnostics averaged over the final 20 recorded measurements during Stage 1. Effective rank and energy concentrations use the batch-level covariance spectrum; cosine similarity measures within-sample token similarity. Prediction error is measured by normalized L1.

Variant	Eff. rank ↑	Top-1 ↓	Top-5 ↓	Cosine ↓	Norm. L1 ↓
EMA	61.28	0.097	0.362	0.071	0.245
No EMA	13.95	0.395	0.674	0.352	0.166

A.4 Attention Analysis Details

The attention probe in Section 4 is produced by our SLIM analysis pipeline. It provides a diagnostic comparison between a Stage-1 pretrained policy and a direct policy trained without masked trajectory prediction. Figure 6 reports the quantitative Stage-1 ablation; the diagnostic rollouts here complement that comparison with qualitative attention analysis.

Analysis pipeline. The paired pipeline captures matched diagnostic rollouts for the pretrained and direct policies and samples representative phases from each sequence. For each frame, it extracts action-to-observation patch attention from layer 15, applies max aggregation over action tokens, and projects the resulting patch scores back to the image plane. The observation row contains the unmodified frames, while dashed boxes identify task-relevant objects for visual reference.

Action-to-patch attention. An action-grounded latent should expose visual factors that matter for the current manipulation rather than preserve only generic visual similarity. We extract action-to-observation attention from the MoT backbone and project the patch-level scores back to the image plane. Figure 7 shows two representative sequences at three manipulation phases. As the interaction progresses, SLIM more consistently follows the task-relevant object, the gripper, and their contact region. Without Stage 1, attention is more diffuse and more often dominated by the robot body or unrelated scene regions. The maps are qualitative probes rather than causal attributions; together with the Stage-1 ablation in Figure 6, they indicate that masked trajectory learning shifts the visual evidence used by the action stream toward manipulation-relevant structure.

A.5 Model Parameterization

Table 6 reports the parameter breakdown of the SLIM configuration used in our LIBERO experiments. The DINOv2-B/14 visual encoder contains 86.58M parameters, while the action model contains 385.56M parameters. The latter is dominated by the 16-layer MoT interaction trunk; its remaining 7.60M parameters comprise the action and state interfaces, learned token embeddings, conditioning projections, and prediction heads. The model size reported in the main paper, 0.47B, corresponds to 472.14M trainable policy parameters. The frozen T5-small language encoder is excluded from the reported 0.47B trainable-policy parameter count.

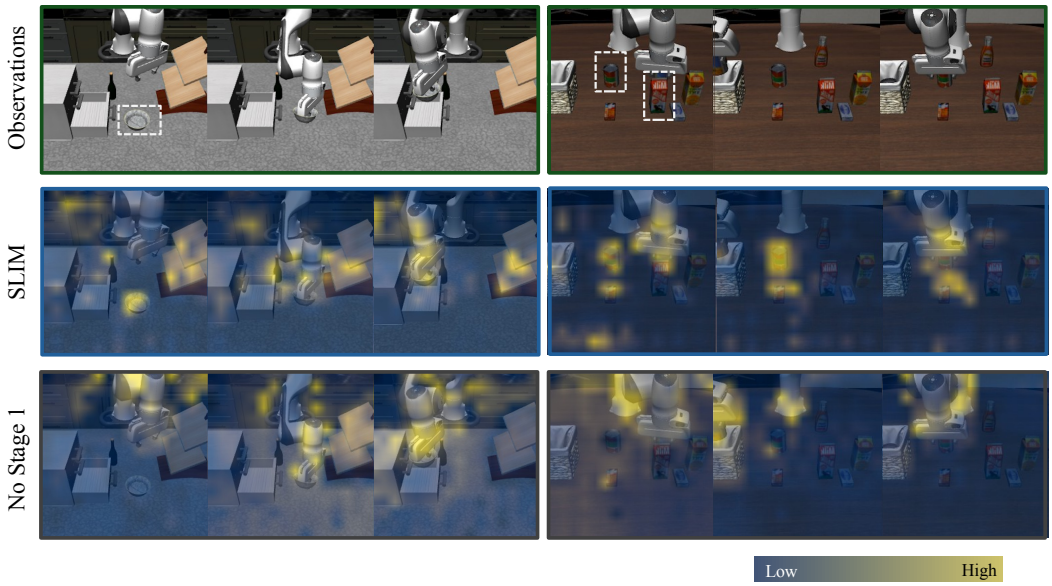


Figure 7: Action-grounding analysis across two representative manipulation sequences. The top row shows observations, with task-relevant objects indicated by dashed boxes; the middle and bottom rows show action-to-observation attention for SLIM and the step-matched policy without Stage 1. Warmer colors denote larger attention. SLIM more consistently follows the manipulated object, gripper, and their interaction region as the task progresses.

Table 6: Parameter breakdown of SLIM. Shares are computed relative to the 472.14M trainable parameters and the component rows are non-overlapping. The frozen EMA target encoder is instantiated only during Stage 1 and is excluded from the reported model size.

Component	Parameters	Trainable share
DINOv2-B/14 visual encoder	86.58M	18.34%
MoT interaction trunk	377.96M	80.05%
Action/state interfaces and prediction heads	7.60M	1.61%
Trainable policy total	**472.14M**	**100.00%**
Frozen EMA visual target encoder	86.58M	–
Stage-1 instantiated total	558.72M	–

For optimizer grouping, 471.58M trainable parameters are subject to weight decay, while 0.56M bias and normalization parameters use zero weight decay. The frozen EMA encoder provides latent prediction targets in Stage 1, receives no gradients, and is not part of the active Stage-2 inference path.

A.6 Implementation Details

Inputs and model configuration. SLIM processes a workspace RGB view and a wrist RGB view, both resized to 224×224. The DINOv2-B/14 visual encoder produces 256 patch tokens per view, yielding $N_z = 512$ current-observation tokens. The MoT uses 16 layers, hidden dimension $d = 768$, and 12 attention heads per stream, together with 512 learned future-slot embeddings of dimension 768. The visual encoder is fine-tuned jointly with the policy, whereas the T5-small language encoder is frozen and its language embeddings are cached. Actions are normalized using per-dataset 1st- and 99th-percentile statistics. Both simulation benchmarks use 7-dimensional actions. Table 7 summarizes their data, training budgets, and action-generation settings.

Table 7: SLIM configurations for LIBERO and CALVIN. The four LIBERO target suites are LIBERO-10, LIBERO-Object, LIBERO-Spatial, and LIBERO-Goal. CALVIN uses the language-annotated ABC training split.

Setting	LIBERO	CALVIN
Stage-1 data	LIBERO-90 + four target suites	ABC
Stage-2 data	Four target suites	ABC
Stage-1 training epochs	3	3
Stage-2 training epochs	40	40
Global batch size (both stages)	128	256
Stage-1 warmup steps	2,000	2,000
Stage-2 warmup steps	5,000	2,000
Action horizon (steps)	8	12
Flow-sampling steps (inference)	4	4

Optimization. We train the simulation policies in BF16 on eight NVIDIA H100 80GB GPUs with gradient clipping at 1.0 and no gradient accumulation. We use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, and weight decay 0.01; bias and normalization parameters receive no weight decay. The action model and visual encoder use learning rates of 10^{-4} and 10^{-5}, respectively, while the remaining trainable parameters use a base learning rate of 2.5×10^{-5}. Learning rates follow cosine decay to a minimum of 10^{-6}. Both the Stage-1 inverse-dynamics objective and the Stage-2 policy objective sample flow times from p_τ by drawing $u \sim \mathrm{Beta}(1.5, 1.0)$ and setting $\tau = (s - \min(u, s))/s$, where $s = 0.999$. Stage 1 uses $\lambda_{\mathrm{IDM}} = 0.125$, $\lambda_{\mathrm{FDM}} = 1$, and EMA momentum 0.999. Stage 2 is initialized from the Stage-1 checkpoint and trained using only the flow-matching policy objective; the EMA target encoder is not retained in Stage 2. The real-world policy uses the same configuration, except that Stage 2 is trained for 20 epochs.

Evaluation protocols. For LIBERO, we evaluate 50 rollouts per task (2,000 rollouts over the four target suites), then evaluate the same checkpoint without adaptation on all 10,030 LIBERO-Plus perturbation cases. CALVIN evaluation follows the standard 1,000 five-instruction chains in the held-out environment D.

A.7 Policy Inference Benchmark Details

Setup. All measurements use one exclusive NVIDIA H100 80GB HBM3 at batch size 1 with two deterministic 224×224 RGB inputs, BF16 precision, and the fixed instruction "do something." The benchmark measures model inference only; model loading, simulator stepping, and RPC are excluded. Each latency run uses 20 warmups followed by 200 synchronized calls, repeated in three fresh processes for 600 pooled measurements. GPU memory is measured in a separate process using `nvidia-smi` at 20 ms intervals during warmup and timed inference, after model, input, and language-cache setup. We report the peak sampled device memory in this window. Model loading, language encoding, and profiler runs are excluded; the text-encoder weights of SLIM and Fast-WAM are not included.

Model configurations and FLOPs. SLIM predicts an 8-step action chunk with 4 sampling steps, $\pi_{0.5}$ predicts a 10-step chunk with 10 sampling steps, and Fast-WAM predicts a 32-step chunk with 10 sampling steps. SLIM and Fast-WAM use precomputed language embeddings, while $\pi_{0.5}$ uses the `pi05_libero` configuration and its native PyTorch policy path. All three models run in PyTorch eager mode, and FLOPs are measured per action chunk using `torch.profiler` with math SDPA. Because native horizons differ, the reported values correspond to one replanning call or action chunk rather than one executed action.

Table 8 reports the absolute measurements for the PyTorch eager comparison used in the main paper.

Table 8: Policy inference cost on one NVIDIA H100 80GB HBM3 at batch size 1 in BF16, using PyTorch eager execution. Latency is the mean over 600 synchronized calls. VRAM is peak sampled device memory during policy inference under the setup described above. Horizons and sampling-step counts are model-specific.

Method	Action horizon	Sampling steps	Latency ↓ (ms/call)	Peak VRAM ↓ (GiB)	FLOPs ↓ (G/chunk)
$\pi_{0.5}$	10	10	193.14	7.94	4714.59
Fast-WAM	32	10	360.64	13.63	2090.07
SLIM	8	4	**60.61**	**4.26**	**490.73**

Backend comparison. A separate JAX/XLA measurement of $\pi_{0.5}$ with the same 10-step action horizon and 10 sampling steps yields 47.66 ms mean latency and 8.71 GiB peak GPU memory during policy inference. This latency is lower than SLIM's PyTorch eager latency; the latency advantage reported above is specific to the PyTorch eager comparison.

A.8 Real-World Evaluation Details

Task-specific OOD settings. Figure 8 shows the nominal setup and all three visual perturbations for each of the five real-world tasks. The background condition replaces the nominal tabletop texture, the lighting condition projects colored light patterns over the workspace, and the distractor condition introduces task-irrelevant objects while preserving the task goal. The same task-condition configurations are used for all evaluated policies.



Figure 8: Task-specific real-world evaluation settings. Columns show the nominal setup and the background, lighting, and distractor perturbations.

Progress-score definitions. Each trial receives a score in $\{0, 0.5, 1\}$ according to the furthest verified task milestone: 0 indicates that neither milestone is reached, 0.5 indicates partial completion, and 1 indicates full completion. For stacking three plates, forming a stack of two plates scores 0.5, while stacking all three scores 1. For placing a carrot into a bowl, grasping the carrot scores 0.5, while placing it into the bowl scores 1. For the toast task, grasping the toast scores 0.5, while placing it onto the plate scores 1. For wiping the whiteboard, grasping the eraser scores 0.5, while completing the wipe scores 1. For stacking blocks, completing the first stacking action scores 0.5, while completing the second and forming the full stack scores 1. Figure 9 illustrates the corresponding initial, partial-progress, and full-completion stages. The main-paper progress score is computed as 100 times the mean trial score.

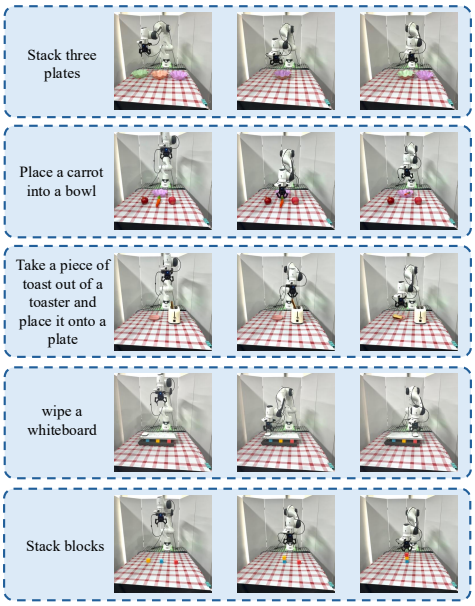


Figure 9: Progress-score milestones for the five real-world tasks. Each row shows the initial state, partial completion (score 0.5), and full completion (score 1).

Per-task results. Figure 10 reports the complete task-wise progress scores under the nominal setting and each OOD perturbation. The rightmost group in each panel is the average across the five tasks and corresponds to the setting-level values summarized in the main paper.

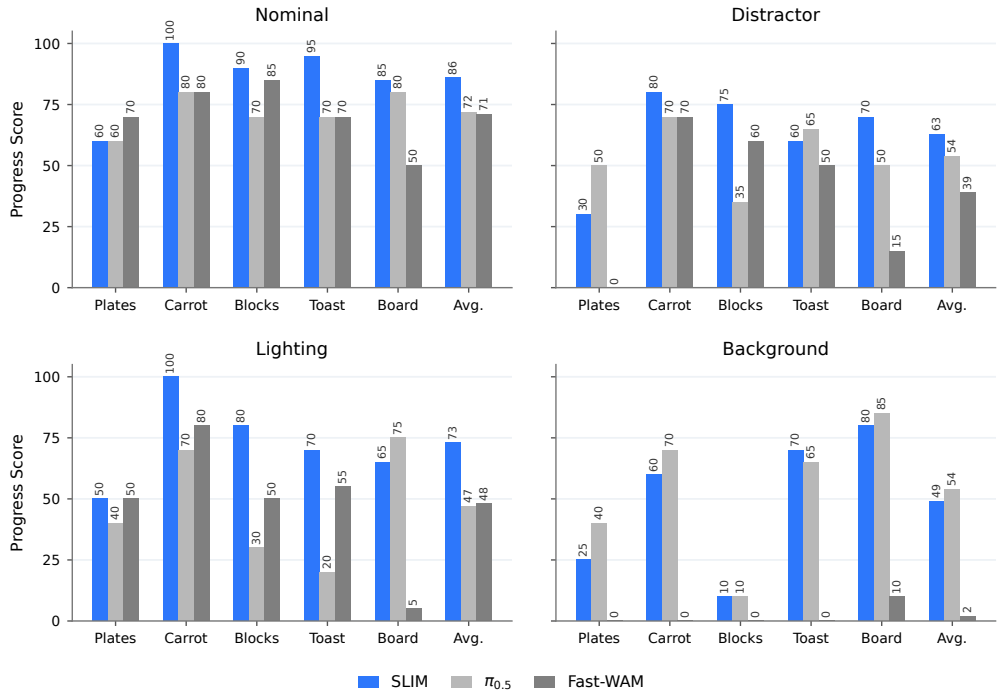


Figure 10: Task-wise real-world progress under the nominal setting and the distractor, lighting, and background perturbations. Each panel reports all five tasks and their average over 10 trials per task.