EDAR: LEARNING ENVIRONMENT-DEPENDENT ACTION REPRESENTATIONS FOR ROBOTIC MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning effective action representations is critical for robotic manipulation, where raw control trajectories are often noisy, redundant, and difficult to model directly. Existing methods mainly encode the structure of the action stream itself, treating the role of actions in the environment as implicit. Yet manipulation is about changing the world: the same action segment can induce different outcomes under different scene contexts, making action semantics inherently environment-dependent. We propose EDAR, an Environment-Dependent Action Representation that grounds action tokens in both executable control structure and expected visual consequences. By coupling motor commands with their environment-conditioned effects, EDAR encourages the learned action space to capture interaction semantics rather than merely command-level patterns. Experiments on simulated and real-robot manipulation benchmarks demonstrate that EDAR improves downstream policy learning, especially in long-horizon manipulation. These results highlight the importance of grounding action representations in executable control structure and environment-conditioned visual change.

1 INTRODUCTION

Raw action trajectories, especially those collected from demonstrations or teleoperation, are often noisy, redundant, and uneven. They contain high-frequency jitter, task-irrelevant micro-corrections, and embodiment-specific control patterns, making them difficult to model directly as a reliable interface for policy learning. This issue becomes more pronounced in manipulation, where policies must generate temporally coherent and physically feasible behaviors over long horizons. Thus, learning effective action representations has become an important problem in robotic manipulation. A good action representation should provide a compact and learnable interface between perception and control, while preserving the information necessary for accurate and robust task execution. The central question is therefore: what kind of action representation is appropriate for robotic manipulation?

Most existing action representations (Pertsch et al., 2025; Wang, 2026) are learned primarily from the action stream itself. They may differ in form, ranging from compact trajectory codes to action tokens or frequency-based representations, but typically share a trajectory-structure-based view: the goal is to preserve or regularize the organization of the trajectory, such as its geometry, temporal variation, smoothness, or predictability. This view helps reduce redundancy and makes actions easier to model, but it remains largely action-only. It describes how a command sequence is structured, while leaving its role in the environment implicit. This is limiting for manipulation, where an action is not merely a motor signal but a means to change the world. The same gripper-closing and rightward-motion segment may be inconsequential in free space, grasp an object under contact, or close a drawer when aligned with a handle, as shown in Fig. 1. Such cases suggest that action semantics are not determined by the command alone, but by the environment-conditioned transition it induces. Therefore, an appropriate action representation for manipulation should not only encode the execution pattern of a command, but also be grounded in its expected effect on the scene.

In this work, we argue that learning environment-dependent action representations requires more than just feeding visual context into an action encoder. Under an action-reconstruction objective alone, the model has a shortcut: it can encode sufficient trajectory information without relying on the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

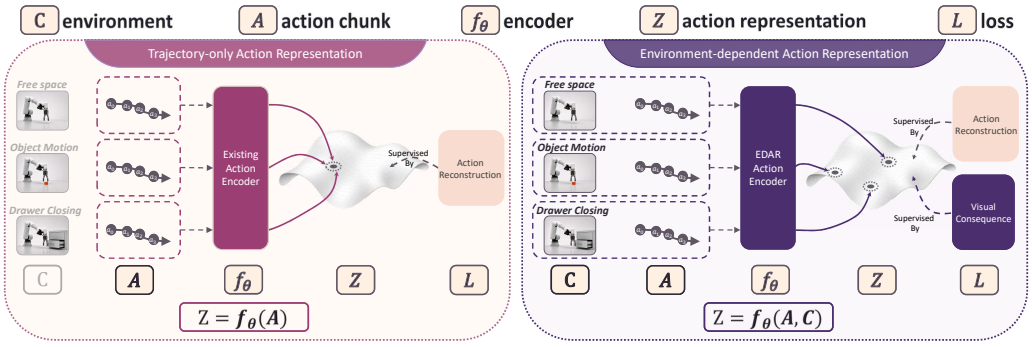


Figure 1: **Conceptual comparison.** The same action segment can induce different effects under different scene contexts, and should therefore not be represented as a context-invariant representation. **Left:** trajectory-only action representation collapses these environment-dependent effects into the same latent action. **Right:** EDAR conditions the action representation on the current scene and grounds it with visual consequences, separating actions according to their context-dependent effects.

visual condition. As a result, the latent space remains organized primarily by command-level similarity: two action segments with similar motor patterns may be mapped nearby even when they play different roles in different scenes. To prevent this collapse, the representation must be constrained by what the action does to the environment. We therefore introduce EDAR, an Environment-Dependent Action Representation that learns latent action tokens from both execution and effect. EDAR preserves the original control trajectory through action reconstruction, while simultaneously requiring the same tokens to model the visual consequence of executing the action under the current scene context. By letting action tokens interact with image tokens in a shared token space, EDAR allows scene context and predicted environmental change to shape the action embedding itself, rather than serving as external conditioning signals. This yields an action representation that is not merely compact, but grounded in interaction semantics.

We conduct extensive experiments in both simulated and real-world manipulation settings. On diverse standard simulation benchmarks including LIBERO (Liu et al., 2023), CALVIN (Mees et al., 2021), and Meta-World (Yu et al., 2020), we compare EDAR-based policies with competitive visuomotor systems. These include strong VQ-based VLA/LLM-style approaches (e.g., FAST, VQ-VLA) as well as continuous-control baselines. Across these benchmarks, EDAR-based policies achieve higher success rates under matched training settings, demonstrating that environment-dependent action representations provide a more effective policy interface than trajectory-centric alternatives. We further validate EDAR on challenging real-world long-horizon manipulation tasks, where it consistently improves over the baseline system under the same training budget. These results establish the empirical advantage of grounding action representations in both executable control structure and environment-conditioned visual consequences.

In summary, our contributions are as follows:
(1) We formulate robotic action representation from an environment-dependent perspective, where action representations explicitly capture both control structure and scene-conditioned visual effects.
(2) We propose **EDAR** (**E**nvironment-**D**ependent **A**ction **R**epresentation), a latent action representation framework that combines action reconstruction with visual consequence modeling in a shared image-action token space.
(3) We show that EDAR improves downstream policy learning on diverse simulated and real-robot manipulation benchmarks, with especially strong gains in challenging long-horizon manipulation.

2 Related Work

Vision-Language-Action Models. Vision-Language-Action (VLA) models provide a paradigm for grounding visual observations and language instructions into robot actions. RT-style systems demonstrated that large sequence models and vision-language priors can be adapted to robotic control (Brohan et al., 2022; 2023), while subsequent models advanced this direction through larger heterogeneous datasets, stronger multimodal backbones, efficient adaptation, and improved action decoders (Team et al., 2024; Kim et al., 2024; Black et al., 2024; Physical Intelligence et al., 2025;

Shukor et al., 2025; Wen et al., 2024; Liu et al., 2024; Gemini Robotics Team, 2025), alongside information exchange across distributed AI components (Shao & Li, 2025; An et al., 2026). Despite these advances, action representation remains a central VLA interface: it defines what the model predicts, how action sequences are structured, and how demonstrations are converted into learnable supervision. Thus, beyond scaling data and backbones, designing better action representations is an important path toward more effective robotic policies. This motivates action representation learning, which aims to replace raw control trajectories with more structured policy interfaces.

Action Representation Learning. Prior work has explored various ways to convert raw control trajectories into more learnable action interfaces. Discrete or tokenized approaches represent actions with frequency-space tokens, spline or control-point tokens, vector-quantized codes, latent skills, or hierarchical token spaces for long-horizon behavior (Pertsch et al., 2025; Zhou et al., 2025; Lyu et al., 2025; Mete et al., 2024; Wang et al., 2025; Liu et al., 2026; Gong et al., 2025; Huang et al., 2026). Continuous approaches instead preserve the structure of action chunks or latent trajectories, enabling temporally coherent prediction and planning in continuous action spaces (Zhao et al., 2023; Wang, 2026; Wu et al., 2026). Although these methods differ in whether the representation is discrete or continuous, they largely share an action-stream-centric view: the representation is learned to preserve, compress, or regularize the structure of the control trajectory itself. As a result, the learned action space is mainly organized by command-level properties such as geometry, smoothness, frequency content, reconstruction fidelity, or multiscale decomposition, while the environment-dependent role of the action remains implicit. EDAR instead focuses on this missing dimension. It uses continuous latent action tokens, but its key contribution is to ground those tokens in the visual consequences induced by actions under the current scene context, allowing environment-conditioned effects to shape the action representation beyond trajectory structure alone.

World Models for Robot Learning. Recent robot learning methods increasingly use future-aware supervision, including future observation prediction, video prediction, and latent world modeling (Li et al., 2025; Zhu et al., 2025; Cen et al., 2025; Kim et al., 2026; Ye et al., 2026). JEPA-style methods perform prediction in latent space to emphasize abstract dynamics over pixel-level details (Assran et al., 2023; 2025; Sun et al., 2026; Vujinovic & Kovacevic, 2025; Maes et al., 2026). These methods demonstrate the value of future prediction for policy learning, planning, and action–world modeling. EDAR is related, but addresses a separate question. Existing world-model approaches typically use actions to condition, generate, or reason about future world states; EDAR instead uses predicted visual consequences to organize the latent action space. This distinction is key: future dynamics are not modeled for their own sake, but are used to make action representations environment-dependent.

3 METHOD

3.1 OVERVIEW

EDAR learns an action representation whose semantics depend on the environment in which the action is executed. At time t, let $A_t = [a_{t+h}]_{h=0}^{H-1}$ denote an action chunk of horizon H. The current and future observations are $I_t = \{I_t^m\}_{m=1}^M$ and $I_{t+H} = \{I_{t+H}^m\}_{m=1}^M$, where M is the number of camera views. We encode A_t into $Z_t = [z_{t,i}]_{i=1}^K \in \mathbb{R}^{K \times d_z}$, where $K \ll H$ is the number of latent action tokens and d_z denotes the latent dimension.

For manipulation, Z_t should be both executable and environment-dependent: it must recover the control trajectory while reflecting the action's role in the current scene. Action-only learning can recover the trajectory but can not distinguish similar motions that correspond to different interactions or consequences across environments. Visual conditioning helps, yet action reconstruction alone still allows the model to ignore the scene and rely on an action-only shortcut.

EDAR therefore addresses these issues by coupling scene-conditioned latent formation with supervision for both execution and effect, as shown in Fig. 2. The current scene participates in latent formation through shared attention among action tokens A_t, visual tokens X_t, and learnable register tokens R. The resulting latent tokens reconstruct A_t while predicting its future visual consequence: action reconstruction preserves executability, while consequence prediction forces Z_t to capture action-induced scene changes. In this way, environmental change is not merely provided as input, but directly shapes the learned action space.

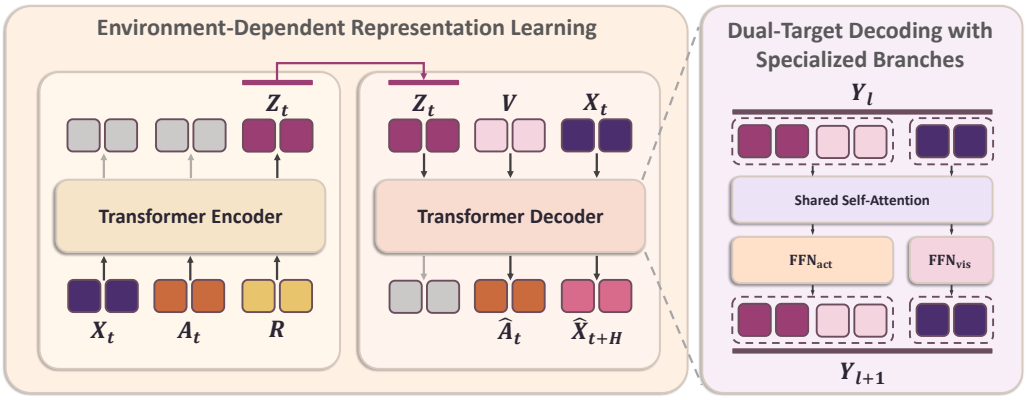


Figure 2: **Overview of Environment-Dependent Action Representation (EDAR).** EDAR encodes the action chunk A_t, current visual tokens X_t, and learnable register tokens R into continuous latent action tokens Z_t. The decoder uses Z_t to reconstruct actions $\hat{A}_t$ from query tokens V and predict future visual tokens $\hat{X}_{t+H}$ from X_t, with shared self-attention and modality-specific $\mathrm{FFN}_{\mathrm{act}}$ and $\mathrm{FFN}_{\mathrm{vis}}$ branches.

3.2 Environment-Dependent Representation Learning

We begin by extracting visual tokens from the current multi-view observations using a pretrained image encoder E_{im}, obtaining $X_t = [E_{\mathrm{im}}(I_t^1); \ldots; E_{\mathrm{im}}(I_t^M)]$. The future visual target is extracted in the same feature space as $X_{t+H} = [E_{\mathrm{im}}(I_{t+H}^1); \ldots; E_{\mathrm{im}}(I_{t+H}^M)]$. We use visual features rather than pixels as the target, so that the supervision emphasizes task-relevant semantic scene change and interaction dynamics instead of low-level appearance reconstruction.

To form the latent action representation, EDAR first projects the action chunk A_t into action tokens and concatenates them with the current visual tokens X_t and K learnable register tokens $R = \{r_i\}_{i=1}^K$. The resulting token sequence is jointly processed by a Transformer Encoder f_θ, with the register outputs taken as the latent action tokens:

$$Z_t = f_\theta([R; A_t; X_t]). \tag{1}$$

The register tokens provide a fixed-size latent interface for variable-length action chunks, while joint self-attention allows the representation to integrate trajectory information with the current scene.

However, context-conditioned latent formation alone is not sufficient. If the only objective is to reconstruct A_t, the representation can still be optimized around trajectory geometry, temporal variation, or other command-level properties. To prevent this shortcut solutions and make the dependence intrinsic, Z_t is trained to explain both the executed action and its consequence. A decoder g_ϕ takes Z_t, X_t, and a set of learnable action queries $V = \{v_i\}_{i=1}^H$, and maps the decoded states of V and X_t to the reconstructed action chunk $\hat{A}_t$ and future visual tokens $\hat{X}_{t+H}$:

$$(\hat{A}_t, \hat{X}_{t+H}) = g_\phi(Z_t, V, X_t). \tag{2}$$

The action reconstruction loss preserves executable structure in the latent representation, while the visual consequence loss encourages it to capture the action's environment-conditioned effect:

$$\mathcal{L}_{\mathrm{act}} = \frac{1}{H} \sum_{h=0}^{H-1} \|a_{t+h} - \hat{a}_{t+h}\|_2^2 . \tag{3}$$

$$\mathcal{L}_{\mathrm{eff}} = \frac{1}{M} \sum_{m=1}^{M} \left(1 - \cos\big(\mathrm{sg}(X_{t+H}^m), \hat{X}_{t+H}^m\big)\right), \tag{4}$$

where $\mathrm{sg}(\cdot)$ denotes stop-gradient, and $\cos(\cdot,\cdot)$ denotes cosine similarity. The stop-gradient operation keeps the visual target space fixed, so that the action representation is shaped by pretrained visual semantics. Thus, the full representation learning objective is

$$\mathcal{L}_{\mathrm{EDAR}} = \mathcal{L}_{\mathrm{act}} + \lambda \mathcal{L}_{\mathrm{eff}}. \tag{5}$$

Both terms prevent distinct degeneracies. Without $\mathcal{L}_{\text{act}}$, the latent may represent scene change without remaining a reliable action interface. Without $\mathcal{L}_{\text{eff}}$, the latent may collapse back to an action-only trajectory code. Their combination makes Z_t both executable and environment-dependent.

3.3 Dual-Target Decoding with Specialized Branches

The two learning signals in EDAR correspond to heterogeneous output spaces: action reconstruction is performed in continuous control space, while visual consequence prediction is performed in pretrained visual feature space. A naive decoder that applies the same token-wise transformation to both targets can introduce optimization conflict, since the two targets require different output structures and levels of abstraction. At the same time, using two separate decoders would weaken the coupling between execution and effect, reducing the ability of visual consequence prediction to shape the same latent action space used for control.

To balance these requirements, EDAR uses a decoder with shared attention and token-specialized feed-forward branches. Let Y^ℓ denote the token states after ℓ decoder layers, initialized as $Y^0 = [\Pi_z(Z_t); V; \Pi_x(X_t)]$. The projections Π_z and Π_x map latent and visual features into the d-dimensional hidden space, with V defined directly in this space. At each decoder layer, tokens interact through a shared self-attention module with an asymmetric causal mask, while the feed-forward update is specialized according to token type. For layers $\ell = 0, \ldots, L-1$, where L is the decoder depth, the updates are:

$$\begin{aligned} \tilde{Y}^\ell &= Y^\ell + \mathrm{Attn}^\ell(Y^\ell), \\ Y^{\ell+1} &= \tilde{Y}^\ell + \left[\mathrm{FFN}^\ell_{\text{act}}(\tilde{Y}^\ell_Z); \mathrm{FFN}^\ell_{\text{act}}(\tilde{Y}^\ell_V); \mathrm{FFN}^\ell_{\text{vis}}(\tilde{Y}^\ell_X)\right]. \end{aligned} \tag{6}$$

Here, $\tilde{Y}^\ell$ is the state after the attention. Subscripts Z, V, and X select the corresponding token positions. After the final layer, separate output heads predict the two targets:

$$\hat{A}_t = o_{\text{act}}(Y^L_V), \qquad \hat{X}_{t+H} = o_{\text{vis}}(Y^L_X). \tag{7}$$

This design keeps the two targets coupled through shared attention, while allowing each target to use a specialized feed-forward transformation. As a result, the decoder supports both control fidelity and consequence modeling without forcing a single output head to serve incompatible prediction spaces. Importantly, the future prediction branch is not used to build a pixel-level world model or to perform rollout-based planning. Its role is to provide an effect-sensitive learning signal that organizes the latent action space according to environment-conditioned consequences.

3.4 Downstream Policy Evaluation

After representation learning, EDAR serves as an action interface for downstream policies. Instead of raw action chunks, the policy predicts latent action tokens Z_t, which are then decoded into executable controls by the learned action decoder.

We instantiate the downstream policy with a flow-matching (Lipman et al., 2022) policy in the EDAR latent space. Given visual observations and language instruction, the policy receives a visuolinguistic condition $\mathbf{c}$. Let $Z_1 = Z_t$ denote the target latent action tokens obtained from EDAR, and sample a Gaussian noise endpoint $Z_0 \sim \mathcal{N}(0, I)$. For interpolation time $\tau \in [0, 1]$, define:

$$Z_\tau = (1-\tau)Z_0 + \tau Z_1. \tag{8}$$

The policy learns a velocity field $v_\psi(Z_\tau, \tau, \mathbf{c})$ in latent space by minimizing:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{\tau,\, Z_0,\, Z_1}\left[\left\| v_\psi(Z_\tau, \tau, \mathbf{c}) - (Z_1 - Z_0)\right\|_2^2\right]. \tag{9}$$

At inference, the policy denoises Gaussian noise into latent action tokens via the learned velocity field. The EDAR decoder then maps the generated latent tokens back to executable robot controls. The downstream policy is not the main contribution of EDAR; rather, it serves as an evaluation protocol for whether the learned action representation provides a more effective policy interface.

Table 1: **LIBERO success rates (%).** success rates (%) across four task suites and average.

Published context						Controlled comparisons					
Method	Spatial	Object	Goal	Long	Avg.	Method	Spatial	Object	Goal	Long	Avg.
Octo (Team et al., 2024)	78.9	85.7	84.6	51.1	75.1	SmolVLA-FAST	87.0	93.0	90.0	68.0	84.5
OpenVLA (Kim et al., 2024)	84.7	88.4	79.2	53.7	76.5	SmolVLA-VQ-VLA	90.0	94.0	86.0	71.0	85.3
VQ-VLA (Wang et al., 2025)	–	–	75.2	60.0	–	SmolVLA (Shukor et al., 2025)	90.0	96.0	92.0	71.0	87.3
Diffusion Policy (Chi et al., 2023)	78.3	92.5	68.3	50.5	72.4	SmolVLA-VAE	90.8	97.6	92.8	75.0	89.1
VLA-4D (Zhang et al., 2025)	93.8	92.8	95.6	86.5	92.2	**SmolVLA-EDAR**	**94.6**	**98.0**	**93.8**	**80.6**	**91.8**
CogACT (Li et al., 2024)	97.2	98.0	90.2	88.8	93.6	π_0-FAST (Pertsch et al., 2025)	96.4	96.8	88.6	60.2	85.5
OmniSAT (Lyu et al., 2025)	94.1	98.7	94.6	86.0	93.4	π_0 (Black et al., 2024)	96.8	98.8	95.8	85.2	94.2
GR00T N1.7 (NVIDIA et al., 2025)	97.7	98.5	97.5	94.4	97.0	**π_0-EDAR**	**98.2**	**99.2**	**96.8**	**94.2**	**97.1**

4 EXPERIMENTS

We design our experiments to test the central hypothesis of EDAR: action representations for robotic manipulation should be environment-dependent. Specifically, we examine whether environment-dependent action representations provide a more effective policy interface than trajectory-centric alternatives. We organize the experiments around three questions:
(1) Environment-dependent advantage. Does EDAR outperform trajectory-centric action representations under matched training settings?
(2) Robustness across settings. Are the gains consistent across different environments, task distributions, and policy backbones?
(3) Long-horizon and real-world stability. Does EDAR improve execution stability in long-horizon and real-world manipulation tasks under the same downstream policy-training budget?

4.1 SIMULATION EXPERIMENTS

We evaluate EDAR on three widely used simulation benchmarks: LIBERO (Liu et al., 2023), CALVIN (Mees et al., 2021), and Meta-World (Yu et al., 2020). LIBERO tests generalization across language-conditioned task suites, CALVIN emphasizes long-horizon control under changing tabletop environment, and Meta-World covers multi-task manipulation across varying difficulty levels. Together, they evaluate the controlled advantage and robustness of environment-dependent action representations across diverse task structures.

4.1.1 EVALUATION ON THE LIBERO BENCHMARK

Setup. We evaluate EDAR on LIBERO (Liu et al., 2023), comprising four suites testing generalization over spatial relations, object categories, task goals, and long-horizon behaviors.

Our controlled study uses SmolVLA (Shukor et al., 2025) as a common backbone, keeping data, preprocessing, and training protocol fixed while varying only the action interface. Within this setting, we compare four action representation schemes: the original SmolVLA action head, trajectory tokenization with FAST and VQ-VLA, and our environment-dependent continuous representation, EDAR. To isolate environment dependence, we include SmolVLA-VAE, which shares EDAR's architecture but removes environment conditioning and visual consequence supervision. We also integrate EDAR into the stronger diffusion-based π_0 model (Black et al., 2024).

Results. Table 1 reports the LIBERO results. Under the controlled SmolVLA setting, EDAR improves the average success rate from 87.3% to 91.8% (+4.5) without changing the backbone, data, or training protocol, with the largest gain on LIBERO-Long from 71.0% to 80.6% (+9.6).

It also outperforms SmolVLA-FAST and SmolVLA-VQ-VLA across all four suites, providing a system-level comparison against representative action-interface alternatives under the same VLA backbone and demonstrating the effectiveness of an environment-dependent continuous latent interface for downstream policy learning. The comparison with SmolVLA-VAE further shows that while a learned latent-action interface is already beneficial, explicitly conditioning it on the environment and supervising it with visual consequences provides additional benefits, improving the average success rate from 89.1% to 91.8% and LIBERO-Long from 75.0% to 80.6%.

Table 2: **CALVIN ABCD→D.** Success rates (%) for completing at least k instructions (of five) and mean completed length (Len.). $*$: pretraining; FSDP: our controlled reproduction.

Published context							Controlled comparisons						
Method	1/5	2/5	3/5	4/5	5/5	Len.	Method	1/5	2/5	3/5	4/5	5/5	Len.
MDT (Reuss et al., 2024)	98.6	95.8	91.6	86.2	80.1	4.52	SmolVLA (Shukor et al., 2025)	92.5	78.9	65.1	53.2	44.7	3.34
MINT-4B (Huang et al., 2026)	97.4	94.2	91.7	88.2	86.1	4.57	**SmolVLA-EDAR**	**94.4**	**84.2**	**75.3**	**67.0**	**58.9**	**3.80**
FLOWER-VLA (Reuss et al., 2025)	98.9	96.7	93.9	90.2	85.5	4.62	FLOWER-VLA (FSDP)	98.0	95.1	91.7	88.2	84.7	4.58
FLOWER-VLA* (Reuss et al., 2025)	99.2	96.9	96.9	92.3	88.3	4.67	**FLOWER-VLA-EDAR**	**99.7**	**98.2**	**96.0**	**92.5**	**88.1**	**4.75**

Table 3: **Meta-World success rates (%).** Avg. is their unweighted mean over four difficulty groups.

Published context						Controlled comparisons					
Method	Easy	Medium	Hard	V. Hard	Avg.	Method	Easy	Medium	Hard	V. Hard	Avg.
Diffusion Policy (Chi et al., 2023)	23.1	10.7	1.9	6.1	10.5	SmolVLA (0.45B, repro.)	88.6	71.8	55.0	62.0	69.4
TinyVLA-H (Wen et al., 2024)	77.6	21.5	11.4	15.8	31.6	**SmolVLA-EDAR**	**89.3**	**76.4**	**78.3**	**78.0**	**80.5**
π_0 (Black et al., 2024)	80.4	40.9	36.7	44.0	50.5	Evo-1 (Lin et al., 2025)	89.2	76.8	77.2	79.2	80.6
RoboTron-Mani (Yan et al., 2024)	85.5	67.7	76.7	87.0	79.3	**Evo-1-EDAR**	**93.2**	**91.8**	**83.3**	**88.0**	**89.1**

EDAR also improves the stronger π_0 backbone, increasing the average success rate from 94.2% to 97.1%, with the largest gain on LIBERO-Long from 85.2% to 94.2% (+9.0), suggesting that EDAR can serve as a plug-in action representation component across different VLA backbones.

Overall, these results support the central claim that effective manipulation action interfaces should preserve executable control structure while grounded in environment-conditioned consequences.

4.1.2 Evaluation on the CALVIN ABCD→D Benchmark

Setup. We further evaluate EDAR on CALVIN ABCD→D (Mees et al., 2021), a long-horizon language-conditioned manipulation benchmark. Each episode executes five instructions without environment reset, making later subtasks depend on earlier scene transitions and providing a natural test of representation robustness over long horizons.

We integrate EDAR into SmolVLA and FLOWER-VLA (Reuss et al., 2025), replacing only the action interface while keeping the remaining architecture and training protocol matched to the corresponding baseline. For FLOWER-VLA, we evaluate EDAR against a matched FSDP reproduction of the baseline. The original published results are reported only as external context, as they use a different distributed-training pipeline.

Results. Table 2 reports the CALVIN results. Under the controlled SmolVLA setting, EDAR improves the average completed sequence length from 3.34 to 3.80. Gains increase as the instruction chain progresses, reaching +1.9, +5.3, +10.2, +13.8, and +14.2 points from 1/5 to 5/5 success, showing stronger benefits as the policy must remain coherent across successive environment transitions.

EDAR also improves the stronger FLOWER-VLA backbone under the controlled FSDP reproduction, increasing the average completed sequence length from 4.58 to 4.75. The gains are consistent across all chain positions, from +1.7 at 1/5 to +3.4 at 5/5.

Overall, CALVIN confirms the long-horizon benefit of environment-dependent action representations. By grounding latent action tokens in expected scene changes, EDAR improves not only local execution but also robustness over extended chains, where earlier actions shape later states.

4.1.3 Evaluation on the Meta-World Benchmark

Setup. We further evaluate EDAR on Meta-World (Yu et al., 2020), grouping tasks into Easy, Medium, Hard, and Very Hard levels and reporting the unweighted average across groups.

We integrate EDAR into SmolVLA and Evo-1 (Lin et al., 2025), replacing only the continuous action interface while keeping data, processing, training, evaluation, and checkpoint selection matched to their reproduced baselines. This isolates the effect of EDAR across differenty policy backbones.

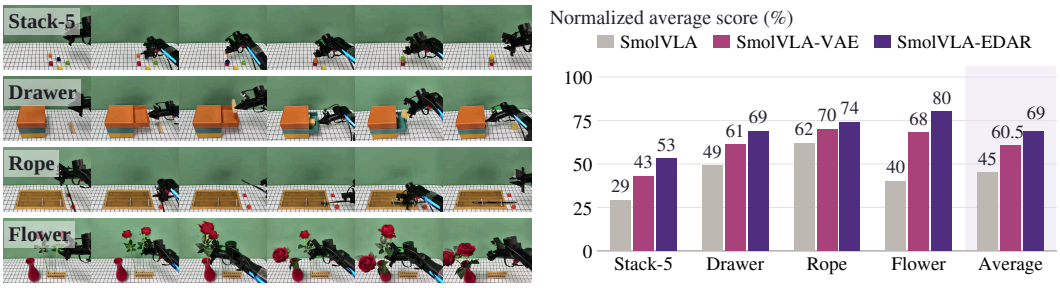


Figure 3: **Long-horizon real-world manipulation. Left:** Six key frames per task show progress from left to right; rows show Stack-5, Drawer, Rope, and Flower. **Right:** normalized task scores and their average over four tasks.

Results. Table 3 reports the Meta-World results. Under the controlled SmolVLA setting, EDAR improves the average score from 69.4 to 80.5 (+11.1), yielding a consistent absolute gain across difficult levels. Gains are modest on Easy tasks (+0.7) but substantially larger on Medium (+4.6), Hard (+23.3), and Very Hard (+16.0), showing greater benefits on more challenging tasks.

On the stronger Evo-1 (Lin et al., 2025) backbone, EDAR improves the average score from 80.6 to 89.1 (+8.5), achieving the highest average performance on this diverse multi-task benchmark. This further supports the effectiveness of EDAR across different benchmarks and policy backbones.

4.2 Long-Horizon Real-World Manipulation

Setup. We evaluate EDAR on four long-horizon real-world manipulation tasks: **Stack-5** (stack four cubes sequentially on a red base cube to form a five-cube tower), **Drawer** (open the top drawer, place a tissue inside, and close it; then open the middle drawer and remove a sponge and close the drawer), **Rope** (thread a rope through two rings sequentially), and **Flower** (pick and place two roses into a vase, one at a time). These tasks involve multiple sequential stages, with later stages depending on scene states produced by earlier actions (Figure 3 left).

Experiments use a single ARX R5 arm with third-person and wrist cameras. We collect 100 teleoperated demonstrations per task and train a task-specific policy. For evaluation, each task is divided into 4–8 sub-steps according to its execution horizon. Each completed sub-step contributes one point, and the total is normalized to a 0–100 score, where 100 indicates full task completion. We report the average score over 10 runs per task.

As the baseline, SmolVLA is trained independently on each task. Against this baseline, we evaluate two latent action representations within SmolVLA: EDAR and VAE. Both representations are pretrained on Open X-Embodiment (O'Neill et al., 2024) following VQ-VLA, then fine-tuned on target-task demonstrations and integrated into SmolVLA. All variants use the same task-specific training budget to isolate the effect of the action interface.

Results Figure 3 right reports the normalized task scores on the four real-world tasks. SmolVLA-EDAR achieves the best performance across all tasks, improving the average score from 45.0% for SmolVLA and 60.5% for SmolVLA-VAE to 69.0%. Compared with SmolVLA, EDAR improves Stack-5, Drawer, Rope, and Flower by +24.0, +20.0, +12.0, and +40.0 points, respectively.

EDAR further improves over SmolVLA-VAE by +8.5 points on average, showing that the gain is not only due to learning a structured latent action representation. With EDAR, the policy better handles changing scene states and reduces compounding errors over long horizons. The consistent improvements across stacking, articulated-object manipulation, and deformable-object manipulation suggest that EDAR is particularly beneficial in real-world sequential manipulation scenarios where intermediate progress directly affects the final outcome.

4.3 Ablation Studies

We analyze the design choices of EDAR in Table 4. Unless otherwise specified, all variants use SmolVLA as the downstream policy backbone and are evaluated on LIBERO under the same data, training schedule, and evaluation protocol as the main experiments.

Table 4: **Ablation study of EDAR on LIBERO.** Panels (a)–(d) ablate environment-conditioned encoding, consequence prediction, decoder design, and visual feature encoders. Gray cells denote Full EDAR; Δ is the change in percentage points relative to Full EDAR.

Action encoding	Avg. ↑	Δ
Full EDAR	**91.8**	0.0
w/o Enc. env. ctx.	90.9	-0.9

(a) **Env.-conditioned encoding.**

Effect supervision	Avg. ↑	Δ
Full EDAR	**91.8**	0.0
w/o future pred.	90.1	-1.7

(b) **Consequence prediction.**

Decoder design	Avg. ↑	Δ
Specialized branches	**91.8**	0.0
Single decoder	90.2	-1.6
Two decoders	88.7	-3.1

(c) **Decoder design.**

Visual encoder	Avg. ↑
VAE baseline	89.1
DINOv3-S	89.9
DINOv3-B	**91.8**
DINOv3-L	**92.0**
I-JEPA	89.2
V-JEPA 2.1	89.7

(d) **Visual feature encoder.**

Environment-conditioned action encoding. Tab. 4a evaluates environment context in action encoding. Removing encoder-side context reduces the average success rate from 91.8% to 90.9%, indicating that scene context provides information beyond the action trajectory alone. This aligns with our motivation that similar trajectories can have different effects across scenes, requiring latent actions to incorporate the current environment.

Consequence prediction supervision. Tab. 4b evaluates future visual prediction as auxiliary supervision. Removing this objective decreases the average success rate from 91.8% to 90.1%, despite retaining environment-conditioned encoding, indicating that environment-conditioned encoding alone is insufficient to make the latent action space effect-aware. Future consequence prediction provides an explicit effect-based signal, encouraging latent action tokens to capture action-induced changes such as object motion, contact outcomes, and task-relevant transitions.

Dual-target decoder design. Tab. 4c compares decoder designs for joint action-effect prediction. EDAR combines shared attention between action and visual tokens with target-specific FFNs for distinct output spaces. A fully shared decoder lowers performance to 90.2%, suggesting interference between the two objectives under shared decoding, while two fully separate decoders further drop performance to 88.7%, indicating that decoupling the two targets weakens action–effect interaction. These results support consequence prediction is more effective when action and effect representations are coupled through shared attention but decoded with target-specific transformations.

Visual feature encoder. Tab. 4d studies the choice of pretrained visual encoder used to construct future feature targets. All variants use the same EDAR setup, downstream SmolVLA policy, and training schedule, and differ only in the visual feature space. The action-only VAE serves as a reference. Within DINOv3 (Siméoni et al., 2025), performance scales with encoder size: DINOv3-S, -B, and -L achieve 89.9%, 91.8%, and 92.0%, respectively, compared with 89.1% for the action-only baseline. This trend suggests that more informative visual targets provide stronger supervision for shaping action latents. JEPA-based targets yield smaller gains: I-JEPA achieves 89.2% and V-JEPA 2.1 reaches 89.7%. This may be because EDAR relies on endpoint dense-feature prediction, where the target space should provide spatially stable local supervision for action-induced scene changes. As visualized in Appendix C, DINOv3 features show clearer object-background separation than V-JEPA2.1 features, making them more effective visual targets for organizing action latents.

5 Conclusion

This work studies action representation learning for robotic manipulation from an environment-dependent perspective. We argue that the meaning of an action should not be defined by its control trajectory alone, but by the consequence it induces in a specific scene. Based on this idea, we propose EDAR, which conditions latent action encoding on the current environment and supervises the latent space with future visual consequences. The resulting representation preserves executability while encouraging action tokens to capture interaction-relevant scene changes. Across simulation and real-world experiments, EDAR provides a more effective action interface for downstream policy learning than trajectory-only representations. Ablations further show that environment-conditioned encoding, future consequence prediction, and the dual-target decoder design each contribute to the final performance. These findings support the broader view that action spaces for embodied agents should be grounded in action–environment interaction rather than only in motor trajectory structure.

AI USE STATEMENT

Generative AI tools were used solely for language editing to improve the clarity, readability, and grammatical quality of the manuscript. They were not used for research methodology, experiment design, implementation, result interpretation, or scientific conclusions. All AI-assisted edits were reviewed and verified by the authors, who take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We provide detailed model architectures, training procedures, hyperparameters, dataset preprocessing, and evaluation protocols in Appendix A. Additional analyses and qualitative results are provided in the subsequent appendices to facilitate reproduction of our experiments.

REFERENCES

Hongjun An, Wenhan Hu, Sida Huang, Siqi Huang, Ruanjun Li, Yuanzhi Liang, Jiawei Shao, Yiliang Song, Zihan Wang, Cheng Yuan, et al. Ai flow: Perspectives, scenarios, and approaches. *Vicinagearth*, 3(1):1, 2026.

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

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, et al. pi_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

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

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023. URL https://arxiv.org/abs/2307.15818.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, et al. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

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. *arXiv preprint arXiv:2303.04137*, 2023. URL https://arxiv.org/abs/2303.04137.

Gemini Robotics Team. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025. URL https://arxiv.org/abs/2503.20020.

Zhefei Gong, Pengxiang Ding, Shangke Lyu, Siteng Huang, Mingyang Sun, Wei Zhao, Zhaoxin Fan, and Donglin Wang. Carp: Visuomotor policy learning via coarse-to-fine autoregressive prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 13460–13470, 2025.

Renming Huang, Chendong Zeng, Wenjing Tang, Jingtian Cai, Cewu Lu, and Panpan Cai. Mimic intent, not just trajectories. *arXiv preprint arXiv:2602.08602*, 2026. URL https://arxiv.org/abs/2602.08602.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, et al. Cosmos policy: Fine-tuning video models for visuomotor control and planning. *arXiv preprint arXiv:2601.16163*, 2026. URL https://arxiv.org/abs/2601.16163.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, et al. CogACT: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024. URL https://arxiv.org/abs/2411.19650.

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model. *arXiv preprint arXiv:2503.00200*, 2025. URL https://arxiv.org/abs/2503.00200.

Tao Lin, Yilei Zhong, Yuxin Du, Jingjing Zhang, Jiting Liu, Yinxinyu Chen, Encheng Gu, Ziyan Liu, Hongyi Cai, Yanwen Zou, et al. Evo-1: Lightweight vision-language-action model with preserved semantic alignment. *arXiv preprint arXiv:2511.04555*, 2025.

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

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

Chaoqi Liu, Xiaoshen Han, Jiawei Gao, Yue Zhao, Haonan Chen, and Yilun Du. OAT: Ordered action tokenization. *arXiv preprint arXiv:2602.04215*, 2026. URL https://arxiv.org/abs/2602.04215.

Jiaming Liu, Mengzhen Liu, Zhenyu Wang, Pengju An, Xiaoqi Li, Kaichen Zhou, et al. RoboMamba: Efficient vision-language-action model for robotic reasoning and manipulation. *arXiv preprint arXiv:2406.04339*, 2024. URL https://arxiv.org/abs/2406.04339.

Huaihai Lyu, Chaofan Chen, Senwei Xie, Pengwei Wang, Xiansheng Chen, Shanghang Zhang, and Changsheng Xu. OmniSAT: Compact action token, faster auto regression. *arXiv preprint arXiv:2510.09667*, 2025. URL https://arxiv.org/abs/2510.09667.

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. URL https://arxiv.org/abs/2603.19312.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *arXiv preprint arXiv:2112.03227*, 2021. URL https://arxiv.org/abs/2112.03227.

Atharva Mete, Haotian Xue, Albert Wilcox, Yongxin Chen, and Animesh Garg. Quest: Self-supervised skill abstractions for learning continuous control. *Advances in Neural Information Processing Systems*, 37:4062–4089, 2024.

NVIDIA, Johan Bjorck, Nikita Cherniadev Fernando Castañeda, 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. In *ArXiv Preprint*, March 2025.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, et al. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. URL https://arxiv.org/abs/2504.16054.

Moritz Reuss, Ömer Erdinç Yağmurlu, Fabian Wenzel, and Rudolf Lioutikov. Multimodal diffusion transformer: Learning versatile behavior from multimodal goals. In *Robotics: Science and Systems*, 2024. URL https://arxiv.org/abs/2407.05996.

Moritz Reuss, Hongyi Zhou, Marcel Rühle, Ömer Erdinc Yagmurlu, Fabian Otto, and Rudolf Lioutikov. FLOWER: Democratizing generalist robot policies with efficient vision-language-action flow policies. *arXiv preprint arXiv:2509.04996*, 2025. URL https://arxiv.org/abs/2509.04996.

Jiawei Shao and Xuelong Li. Ai flow at the network edge. *IEEE Network*, 40(1):330–336, 2025.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

Oriane Siméoni, Huy V. Vo, et al. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025. URL https://arxiv.org/abs/2508.10104.

Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, et al. VLA-JEPA: Enhancing vision-language-action model with latent world model. *arXiv preprint arXiv:2602.10098*, 2026. URL https://arxiv.org/abs/2602.10098.

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.

Aleksandar Vujinovic and Aleksandar Kovacevic. ACT-JEPA: Joint-embedding predictive architecture improves policy representation learning. *arXiv preprint arXiv:2501.14622*, 2025. URL https://arxiv.org/abs/2501.14622.

Junming Wang. Latentvla: Taming latent space for generalizable and long-horizon bimanual manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18593–18601, 2026.

Yating Wang, Haoyi Zhu, Mingyu Liu, Jiange Yang, Hao-Shu Fang, and Tong He. Vq-vla: Improving vision-language-action models via scaling vector-quantized action tokenizers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. arXiv:2507.01016.

Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Kun Wu, Zhiyuan Xu, et al. TinyVLA: Towards fast, data-efficient vision-language-action models for robotic manipulation. *arXiv preprint arXiv:2409.12514*, 2024. URL https://arxiv.org/abs/2409.12514.

Songwei Wu, Zhiduo Jiang, Guanghu Xie, Wandong Sun, Hong Liu, and Yang Liu. CoLA-Flow policy: Temporally coherent imitation learning via continuous latent action flow matching for robotic manipulation. *arXiv preprint arXiv:2601.23087*, 2026. URL https://arxiv.org/abs/2601.23087.

Feng Yan, Fanfan Liu, Liming Zheng, Yufeng Zhong, Yiyang Huang, Zechao Guan, Chengjian Feng, and Lin Ma. Robotron-mani: All-in-one multimodal large model for robotic manipulation. *arXiv preprint arXiv:2412.07215*, 2024.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026. URL https://arxiv.org/abs/2602.15922.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 2020. URL https://proceedings.mlr.press/v100/yu20a.html.

Haochen Zhang et al. VLA-4D: Embedding 4d awareness into vision-language-action models for spatiotemporally coherent robotic manipulation. *arXiv preprint arXiv:2511.17199*, 2025. URL https://arxiv.org/abs/2511.17199.

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

Hongyi Zhou, Weiran Liao, Xi Huang, Yucheng Tang, Fabian Otto, Xiaogang Jia, et al. BEAST: Efficient tokenization of b-splines encoded action sequences for imitation learning. *arXiv preprint arXiv:2506.06072*, 2025. URL https://arxiv.org/abs/2506.06072.

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. *arXiv preprint arXiv:2504.02792*, 2025. URL https://arxiv.org/abs/2504.02792.

A Implementation Details

This section provides details on the architecture and training procedures across all benchmarks. We describe the details as follows.

A.1 Architecture

Overview. EDAR is implemented as an 8-layer Transformer encoder paired with an 8-layer Transformer decoder, each with a hidden size of 512, 8 attention heads, and feed-forward networks with a $4\times$ expansion ratio. We use LayerNorm as the normalization layer for all Transformer blocks. The image encoder is instantiated as DINOv3-Base and is used to extract visual features from image observations, as shown in Fig. A. The dimensionality of each latent action token is set equal to the original action dimension for the corresponding robot embodiment.

Visual feature encoder and feature space. EDAR uses the current multi-view observation as visual context and predicts both the action chunk and a future visual-context target. Standard EDAR uses DINOv3-Base patch features; the DINO encoder is used as a frozen feature extractor in the action representation implementation, providing a common feature space for current and future observations. The decoder predicts actions and future DINOv3 patch features. The image prediction loss is implemented as a cosine-distance loss between predicted and target DINO features, scaled by the configured image supervision weight, with adaptive scaling as described below.

Encoder. The encoder uses a bidirectional self-attention mask over all input tokens, including current visual tokens, action tokens, and learnable register tokens. This design allows every token to access the entire action chunk and the visual environment, so information can be exchanged across action timesteps and camera views. The register tokens can aggregate temporal information across the full action sequence while interacting with environment context, so that the latent action tokens are formed from both executable action structure and environment-dependent information.

Decoder. The decoder uses an asymmetric attention mask tailored to the heterogeneous reconstruction targets of its action and visual branches. For future visual feature prediction, visual queries are allowed to attend to the full decoder token set, including latent tokens and action-token states. This enables the visual branch to predict the endpoint scene consequence from complete action and latent information. For action reconstruction, action queries follow a causal reconstruction mask: each action position can attend to the latent action tokens and its available causal action context, but is prevented from attending to future action positions and visual prediction tokens. This preserves the causal dependency structure of action reconstruction and prevents future visual tokens from serving as a shortcut. As a result, the decoder couples action reconstruction and visual consequence prediction through shared latent conditioning, while maintaining a clean generation path from action embedding tokens to executable controls.

Adaptive visual supervision. The coefficient λ in the main text is a simplified notation. In practice, the effective visual loss coefficient is $\lambda_{\text{eff}} = \lambda\, w_{\text{adapt}}$, with $w_{\text{adapt}} = \|\nabla_{\theta_s}\mathcal{L}_{\text{act}}\|_2 / \|\nabla_{\theta_s}\mathcal{L}_{\text{eff}}\|_2$. The adaptive factor w_{adapt} balances the two objectives by scaling the visual supervision according to their gradient magnitudes on the shared decoder parameters, preventing either loss from dominating optimization. Here, λ is the base weight in Table A, and θ_s denotes the parameters at the nearest decoder block shared by the two losses.

A.2 Training Pipeline

The training adopts a two-stage pipeline for representation learning and downstream control. In the first stage, EDAR is trained to capture environment-dependent, continuous latent representations of action trajectories. In the second stage, the pretrained EDAR is embedded into a policy model, and this policy is trained for downstream control tasks while EDAR's parameters are kept frozen. We use fixed random seeds to ensure reproducibility.

LIBERO. For LIBERO experiments, we use the cleaned and reformatted dataset released by the HuggingFace team following the preprocessing pipeline of OpenVLA (Kim et al., 2024). The

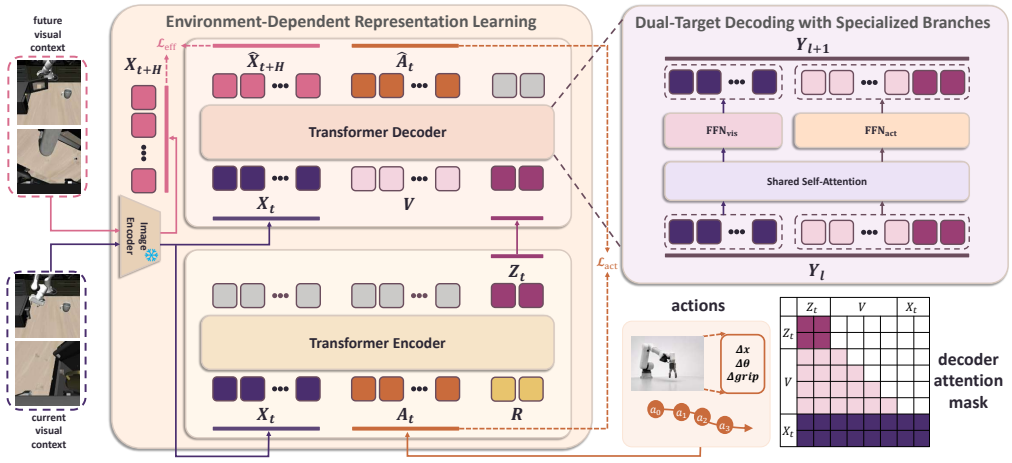


Figure A: **Detailed EDAR mechanism.** EDAR forms latent action tokens through full-attention interaction among current visual tokens, action tokens, and learnable register tokens, then decodes them to reconstruct actions and predict future visual features. The bottom-right panels illustrate the decoder asymmetric mask, where visual prediction tokens can attend to the full token set, while action reconstruction remains causal and independent of visual prediction tokens.

Table A: **Training recipes for EDAR representation learning.**

Parameter	Simulation			Real-world
	LIBERO	CALVIN	Meta-World	
Latent action dim	7	7	4	7
Image supervision weight λ	0.2	0.1	0.2	0.2
Training steps	100k			100k
Batch size	64			128
Chunk size	16			20
Latent action tokens K	4			4
Learning rate schedule	$10^{-4} \rightarrow 10^{-5}$			$2^{-4} \rightarrow 2^{-5}$
Optimizer	Adam			Adam
Optimizer betas	$(0.9,\ 0.99)$			$(0.9,\ 0.99)$

dataset removes no-op frames and failed episodes, and contains 1,693 expert demonstrations converted to the LeRobot format, with paired visual observations and action trajectories. We use the same dataset version as the SmolVLA baseline to ensure a fair comparison.

EDAR is first trained in a representation-learning stage using the training setting in Table A, with action reconstruction and future visual consequence prediction as the learning objectives. We then integrate the pretrained EDAR module into SmolVLA by replacing the original action interface with EDAR latent action tokens. The downstream policy is trained on the same LIBERO dataset for 100k steps with batch size 64 and chunk size 16. The learning rate follows a cosine decay schedule from $1.5\text{e}{-4}$ to $5\text{e}{-6}$. During evaluation, the policy predicts 16 future actions at each step and executes the first 8 actions before generating a new chunk from updated observations.

CALVIN. For CALVIN, we use the official CALVIN ABCD→D benchmark converted into the LeRobot format. CALVIN is a long-horizon language-conditioned manipulation benchmark built from human-teleoperated demonstrations in a tabletop environment (Mees et al., 2021). The dataset contains four environment splits, denoted A, B, C, and D, each corresponding to a different tabletop scene configuration. In the standard ABCD→D protocol, policies are trained on splits A–D and evaluated on the held-out evaluation sequences in environment D.

We first train EDAR on the CALVIN dataset using the recipe in Table A. The EDAR module is integrated into SmolVLA as a frozen action interface, and the downstream policy is trained on the same dataset for 100k steps with batch size 64 and chunk size 16. The learning rate follows a cosine schedule decaying from $1.5\mathrm{e}{-4}$ to $5\mathrm{e}{-6}$. During evaluation, the policy predicts 16 actions and executes the first 8 at each step.

We additionally integrate the same CALVIN-trained EDAR module into FLOWER-VLA (Reuss et al., 2025). For controlled comparison, EDAR only replaces the action representation interface, while all FLOWER-VLA training hyperparameters and evaluation settings are kept unchanged, including the training budget, batch size, optimizer and scheduler settings, denoising architecture, and evaluation action horizon. Following the original setup, the policy is trained for 40k steps across 4 GPUs with batch size 8 per GPU, and the action horizon is set to 10 during evaluation. Different from the original DDP implementation, our reproduction uses FSDP for distributed training.

Meta-World. For Meta-World, we follow the standard MT50 evaluation setting used in SmolVLA (Shukor et al., 2025) and evaluate on 50 manipulation tasks grouped into Easy, Medium, Hard, and Very Hard difficulty categories.

We first train EDAR on Meta-World using the representation-learning recipe in Table A, with action reconstruction and future visual consequence prediction as the learning objectives. The original SmolVLA paper reports Meta-World results, but omits some Meta-World-specific hyperparameters needed for exact reproduction. We therefore build a controlled SmolVLA-Meta-World baseline under our verified training recipe. The baseline loads the pretrained VLM weights and uses a four-dimensional state and action interface, chunk size 16, action horizon 8, 100k scheduler decay steps, and a cosine learning-rate schedule from $1.5\mathrm{e}{-4}$ to $5\mathrm{e}{-6}$. We then integrate the pretrained EDAR module into the same reproduced SmolVLA pipeline by replacing the original action interface with EDAR latent action tokens. All other settings, including data processing, training schedule, learning-rate schedule, evaluation code, and checkpoint selection, are kept identical.

Real-World Experiments. For real-world EDAR representation learning, we adopt a two-stage training procedure. In the first stage, EDAR is pretrained on 10 Hz, 15 Hz, and 20 Hz subsets from the OpenX dataset (O'Neill et al., 2024), where samples are drawn uniformly across individual datasets. In the second stage, EDAR is further trained on our self-collected tele-operated real-world data. Both stages follow the same real-world training recipe in table A.

For downstream policy learning, the pretrained EDAR module is integrated into SmolVLA by replacing the original action interface with EDAR latent action tokens. The downstream policy is trained on the same self-collected task data for 100k steps with batch size 64 and chunk size 20. The learning rate follows a cosine decay schedule from 1.5×10^{-4} to 5×10^{-6}. The SmolVLA baseline uses the same policy training setup.

B Environment-dependent Structure of Action Latent Space

Figure B illustrates the environment-dependent structure of the action latent space by comparing action chunks with similar command-level motion but different scene contexts. The goal is to examine whether EDAR assigns different latent representations to similar motor patterns when the environment changes their expected consequences.

We construct the visualization from 16-step action chunks in the LIBERO demonstration dataset. For each displayed pair, the two chunks have similar local command structure, as shown by the near-overlapping action trajectories, but are executed in different visual contexts. We compare an environment-independent action encoder with EDAR. The baseline encodes each action chunk without the current observation, while EDAR encodes the same chunk together with its corresponding current visual observation. We then project the latent representations into two dimensions with t-SNE. Gray points show sampled background chunks, and colored points highlight the matched pairs, using consistent colors across the latent maps and example panels.

Each middle panel shows one matched pair from two LIBERO episodes. The start and end observations show the scene before and after the 16-step horizon, illustrating that similar motions can correspond to different environment-conditioned consequences. The reported pairwise distance is

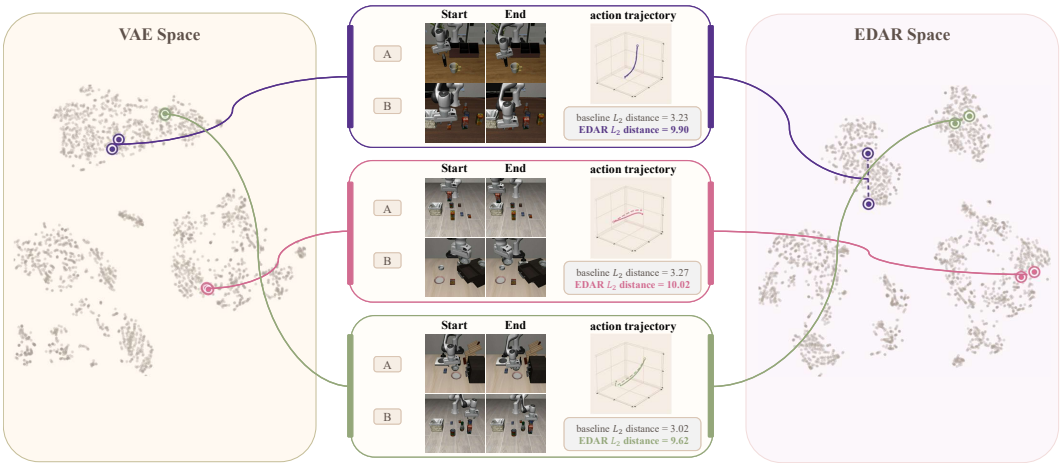


Figure B: **Environment-dependent structure of action latent space.** We compare the environment-independent VAE and EDAR on the same three matched action pairs from LIBERO. Each middle panel shows one pair from two different episodes, with start/end observations and an action trajectory indicating similar short-horizon command structure. In the environment-independent baseline latent map, each pair remains close, while in the EDAR latent map the same pair becomes more separated under different visual contexts. The reported L_2 distances measure the separation between the two highlighted points in their raw latent spaces.

computed as the Euclidean distance between the two corresponding representations in the latent space, before the two-dimensional projection used for visualization.

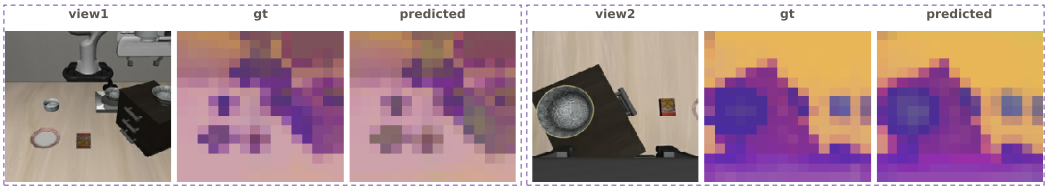
Across the three examples, the environment-independent encoder places each pair almost on top of each other, reflecting their similar command-level motion. In contrast, EDAR separates the same pairs more clearly when they occur under different scene contexts. This suggests that EDAR does not merely encode short-horizon motor patterns, but allows scene context to modulate the latent action representation. Similar action chunks can therefore occupy different regions of the latent space when they are associated with different expected effects on the environment.

C Visualization of Visual Context Prediction

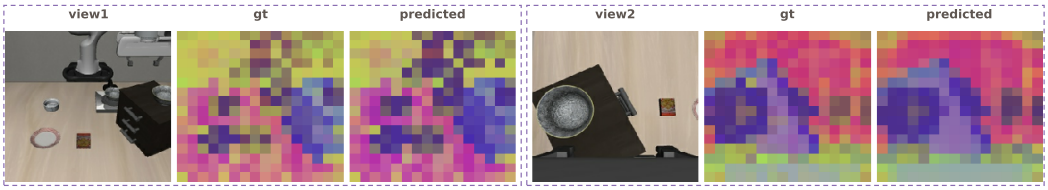
Figure C visualizes future visual-context feature prediction for a randomly selected LIBERO action segment under the standard DINOv3-based EDAR checkpoint and a V-JEPA2.1 variant. Since the ground truth is a feature field rather than an RGB reconstruction, each example shows the raw future image together with palette-stylized PCA projections of the ground-truth and predicted future patch features. The left and right sides of each strip correspond to the two camera views used as visual context: the main camera and the wrist camera, respectively.

The visualization helps explain why DINOv3 features serve as a stronger visual target for EDAR. Since EDAR predicts an endpoint future feature field rather than an RGB image or a full video rollout, the target feature space should provide spatially stable and locally interpretable supervision. In Fig. C, the DINOv3 projections show clearer separation between task-relevant objects, robot parts, and background regions. The predicted DINOv3 feature field also preserves much of this region-level layout, suggesting that the auxiliary loss provides a clean signal for associating action latents with object-centric scene changes.

By contrast, the V-JEPA2.1 projections appear more locally fragmented, with weaker separation between foreground objects and background regions. Although V-JEPA2.1 is designed for image-video representation learning, its representation space is less aligned with EDAR's static endpoint dense-feature regression objective. This mismatch may make the visual consequence loss less informative for organizing action latents.



(a) DINOv3.



(b) V-JEPA2.1.

Figure C: **Visualization of visual context prediction.** We compare the predicted future feature maps when EDAR is supervised with DINOv3 and V-JEPA2.1 features on the same black-bowl segment. Each subfigure corresponds to one target feature space. Within each subfigure, we show both the main-camera view and the wrist-camera view; for each view, the panels are ordered as the future RGB observation, ground-truth future feature map, and predicted future feature map. The colored maps are PCA projections of patch-level visual features in the corresponding encoder space.

D Qualitative Results

We provide qualitative visualizations of successful rollouts across all benchmarks. In each figure, every row shows one task rollout, with the leftmost frame depicting the initial state and the rightmost frame depicting the final state.

D.1 Qualitative Results on LIBERO

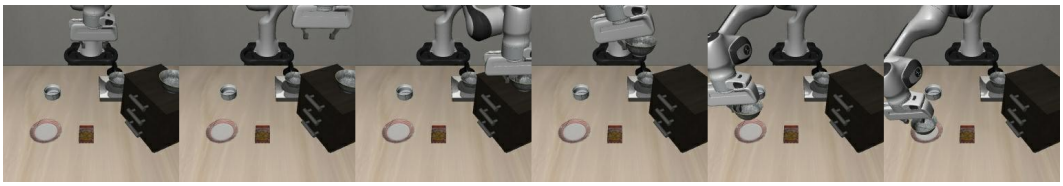
As illustrated in Figure D, we show sampled rollout frames from the four LIBERO task suites: *Spatial*, *Object*, *Goal*, and *Long*. Each row corresponds to one task suite, where we uniformly sample six frames from one representative episode.

D.2 Qualitative Results on CALVIN

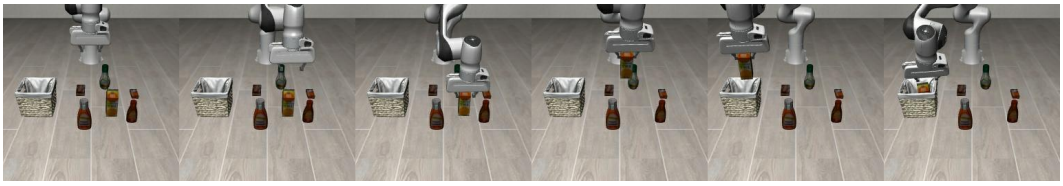
Figure E visualizes successful multi-step CALVIN evaluation sequences. Each row is formed from six square main-view frames: the initial frame of the sequence followed by one late frame from each of the five successful subtasks in the same rollout sequence.

D.3 Qualitative Results on Meta-World

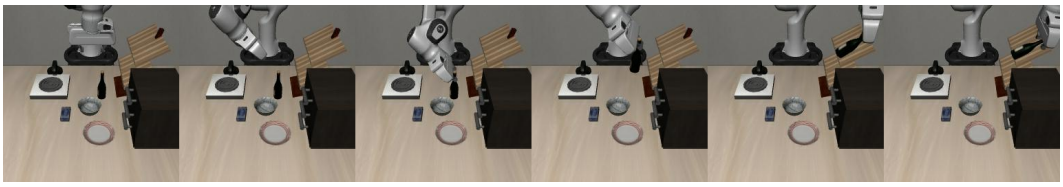
Figure F visualizes representative Meta-World MT50 tasks. Each row corresponds to one task and contains six uniformly sampled frames from one episode.



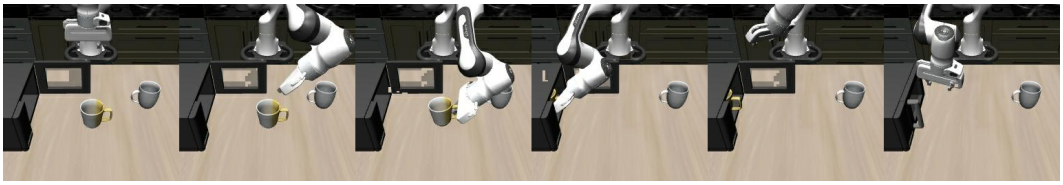
(a) Spatial: pick up the black bowl on the wooden cabinet and place it on the plate.



(b) Object: pick up the orange juice and place it in the basket.



(c) Goal: put the wine bottle on the rack.



(d) Long: put the yellow and white mug in the microwave and close it.

Figure D: **Qualitative results on LIBERO tasks.** Representative examples include tasks from Spatial, Object, Goal, and Long.

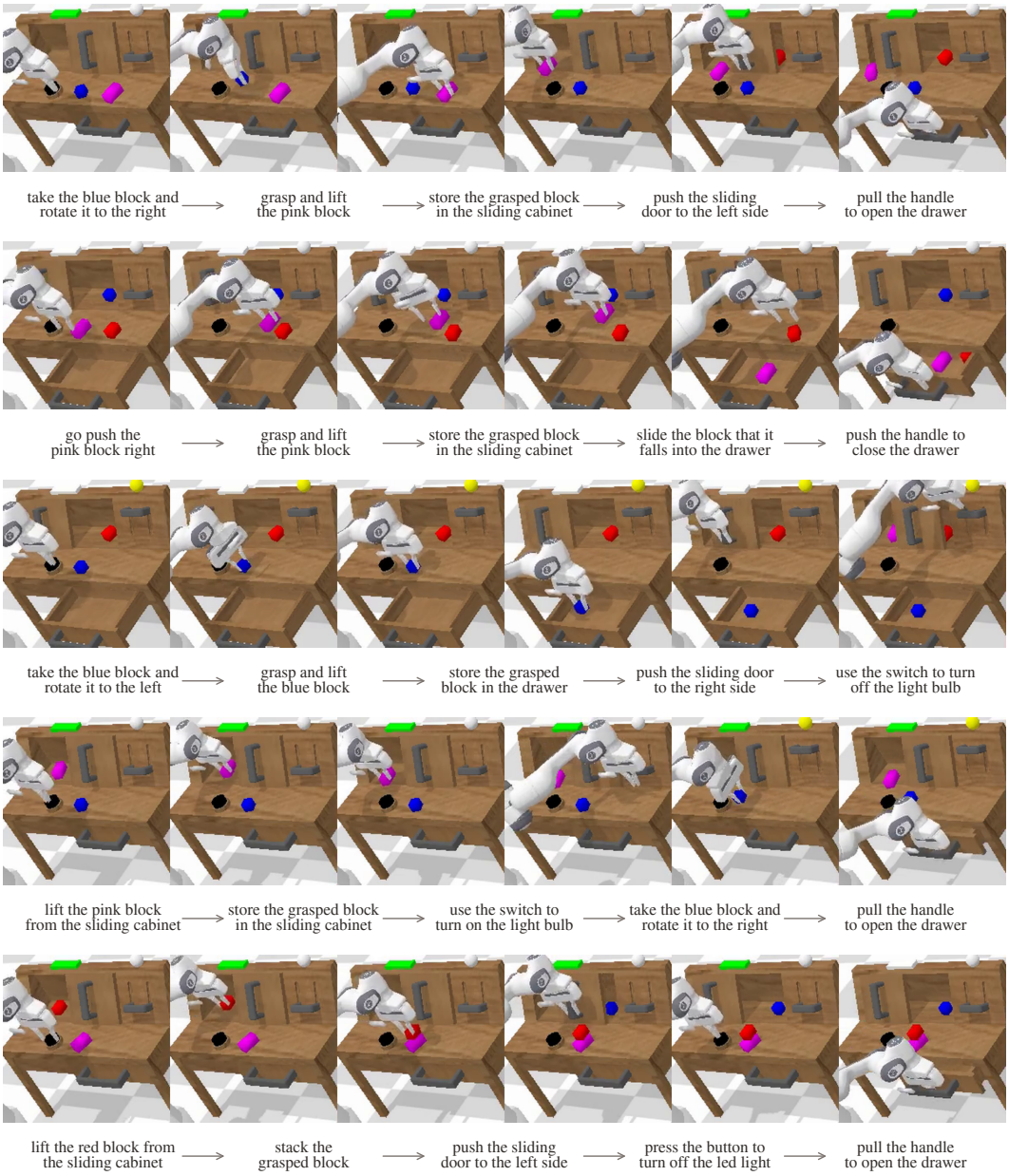
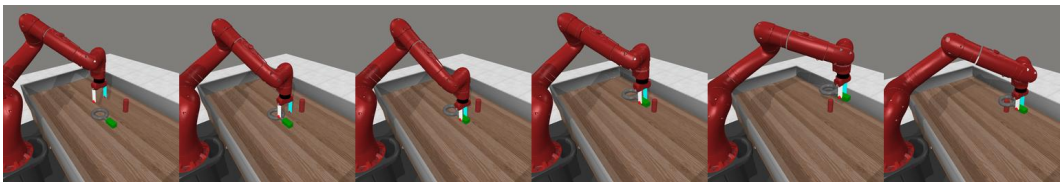
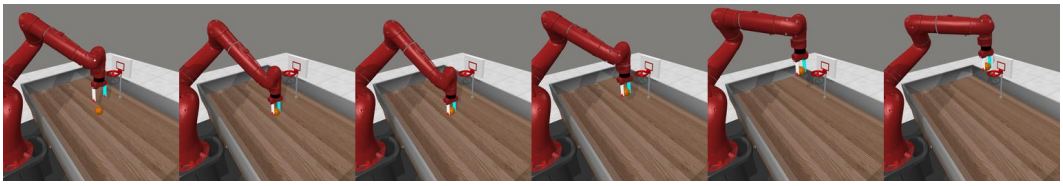


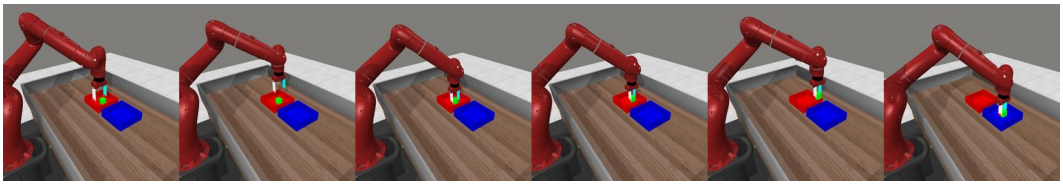
Figure E: **Qualitative results on CALVIN.** Each row shows one successful 5-step task sequence reconstructed from the local CALVIN evaluation videos using only the main camera view.



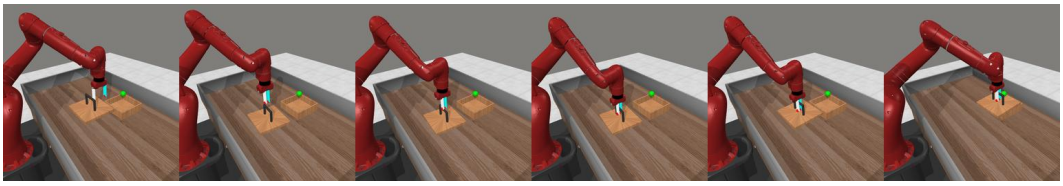
(a) Pick up a nut and place it onto a peg.



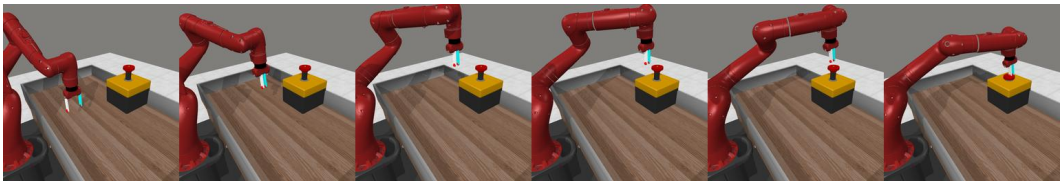
(b) Dunk the basketball into the basket.



(c) Grasp the puck from one bin and place it into another bin.



(d) Grasp the cover and close the box with it.



(e) Press a button from the top.

Figure F: **Qualitative results on Meta-World.** Representative examples include peg, basketball, puck-transfer, box-closing, and button-press tasks from the MT50 benchmark.