

ProRet-VLA: Bridging Prospection and Retrospection for Long-Horizon Robotic Manipulation

Enhua Cheng, Xiaoyuan Fang, Yuxuan Wang, Piji Li[†]

College of Artificial Intelligence,

Nanjing University of Aeronautics and Astronautics, Nanjing, 211106, China

The Key Laboratory of Brain-Machine Intelligence Technology,

Ministry of Education, Nanjing, 211106, China.

{ehcheng,pjli}@nuaa.edu.cn

Abstract

Vision-Language-Action (VLA) models map visual observations and language instructions to low-level control actions, but their reliance on the current observation often limits temporal continuity in long-horizon manipulation. We propose ProRet-VLA, a temporal reasoning framework that links prospection and retrospection through predictive consistency. ProRet-VLA maintains compact episodic memories and predicts future observations from temporally enriched context. Rather than using future prediction only as an auxiliary target, we use its prediction error as a self-supervised deviation signal to train a lightweight Deviation Predictor. During inference, the predicted deviation conditions token-wise fusion between current context and retrieved memory, allowing memory usage to vary across decision stages without accessing future observations. Experiments on CALVIN ABC-D and LIBERO show that ProRet-VLA improves long-horizon task completion, achieving an average length of 4.50 on CALVIN and a 96.7% average success rate on LIBERO. Real-world experiments on two long-horizon manipulation tasks further indicate improved temporal progress tracking over strong VLA baselines.

1 Introduction

Vision–Language–Action (VLA) models have recently achieved strong generalization in robotic manipulation (Zitkovich et al., 2023; Kim et al., 2024; Li et al., 2024a; Black et al., 2024; Intelligence et al., 2025), leveraging pre-trained Vision–Language Models (VLMs) (Bai et al., 2023; Achiam et al., 2023; Karamcheti et al., 2024) to directly map visual observations and language instructions to actions. Empowered by internet-scale pre-training, they transcend simple pattern matching to generalize across diverse instructions and unstructured environments. Despite this progress,

[†]Corresponding author.

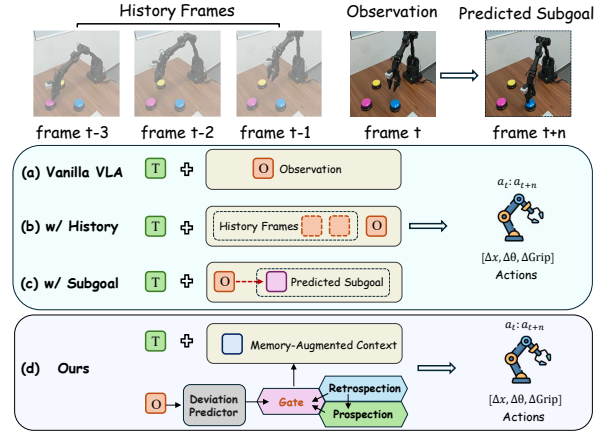


Figure 1: (a) Vanilla VLA directly maps visual observations and language instructions to actions. (b) VLA with history frames integrates a memory mechanism of past observations. (c) VLA with subgoal explicitly predicts a future image as an intermediate visual reasoning step prior to action generation. (d) Our proposed **ProRet-VLA** bridges prospection and retrospection through a deviation-conditioned gate, using predictive consistency to regulate how retrieved history is fused with the current context for action generation.

their limited temporal modeling capacity causes visual short-sightedness (Lin et al., 2025), degrading performance on long-horizon tasks.

Vanilla VLA models (Kim et al., 2024, 2025; Black et al., 2024) rely solely on current observations (Fig.1 (a)), causing temporal short-sightedness that fragments trajectories in long-horizon tasks. To alleviate this, recent works pursue two isolated directions: One incorporates episodic memory (Sridhar et al., 2025; Li et al., 2025b; Koo et al., 2025; Shi et al., 2025) (Fig.1 (b)), allowing agents to leverage past experiences for current decision-making. The other focuses on future foresight, employing generative world models or video generation techniques (Zhao et al., 2025; Liu et al., 2025b; Tian et al., 2025; Zhang et al., 2025) (Fig.1 (c)), to anticipate future dynamics and guide proactive planning. However, this isolation overlooks their interdependence: purely retrieval-based methods lack active foresight, while isolated world

models are conversely prone to hallucination without sufficient historical grounding. Consequently, agents fail to integrate past context with future anticipation, limiting their robustness.

To address these issues, we propose ProRet-VLA (Prospection and Retrospection for VLAs), a temporal reasoning framework that connects prospective prediction with retrospective memory through predictive consistency. Inspired by the role of prediction error in predictive coding (Friston, 2010), our key idea is not to simply attach a memory module to a world model, but to use future prediction error as a self-supervised signal for regulating when retrieved history should influence action generation. As illustrated in Fig. 2, ProRet-VLA first stores historical interactions as compact latent memories, avoiding the cost of retaining raw observations while preserving task-relevant temporal context. It then performs memory-grounded prospection, where learnable Prospection Tokens predict future observation features from both the current context and retrieved history. The resulting prediction error provides a training-time deviation target, which supervises a lightweight Deviation Predictor. During inference, the predicted deviation conditions a token-wise memory fusion gate, allowing the policy to adaptively balance current observations and retrieved memories across different decision stages without accessing future observations. In this way, predictive consistency serves as a bridge between prospection and retrospection: future prediction is not merely an auxiliary objective, but a mechanism for learning how retrospective memory should be used in long-horizon control.

The contributions of our work are summarized as follows:

- We propose ProRet-VLA, a unified temporal reasoning framework for robust long-horizon VLA control that connects prospective future prediction with retrospective episodic memory through predictive consistency.
- We introduce a deviation-conditioned memory fusion mechanism. Future prediction error provides a self-supervised training signal for a lightweight Deviation Predictor, whose output regulates token-wise memory fusion during inference without requiring future observations.
- Extensive experiments on the CALVIN (Mees et al., 2022), LIBERO (Liu et al., 2023) benchmarks and real-world long-horizon tasks, demonstrate that ProRet-VLA achieves substantial per-

formance. Our method exhibits superior temporal coherence and robustness, significantly outperforming existing baselines in long-horizon manipulation tasks.

2 Related Work

2.1 Vision-Language-Action Models

Inspired by recent VLMs (Achiam et al., 2023; Bai et al., 2023) and large-scale datasets (O’Neill et al., 2024; Ebert et al., 2021; Khazatsky et al., 2024), VLA models are trending in robot learning. Models like RT-2 (Zitkovich et al., 2023) and OpenVLA (Kim et al., 2024) discretize actions for autoregressive prediction, whereas approaches like π_0 (Black et al., 2024), CogACT (Li et al., 2024a), and HybridVLA (Liu et al., 2025a) adopt diffusion policies (Chi et al., 2025) to generate continuous trajectories capturing diverse behaviors. However, most VLA models remain Markovian, reasoning solely on current observations. This temporal short-sightedness ignores dynamic task evolution, causing incoherent decisions in long-horizon tasks.

2.2 Temporal Reasoning in Robotics

Though extensively explored in video understanding, temporal modeling remains under-utilized in VLA architectures. Existing approaches typically employ one-sided reasoning. One line incorporates historical context to ground decisions (e.g., RoboVLMs (Li et al., 2024b), Octo (Ghosh et al., 2024), TraceVLA (Zheng et al., 2024), MemoryVLA (Shi et al., 2025)). However, these passive methods lack generative prospection to anticipate dynamics. Conversely, another line predicts future visual subgoals (e.g., CoT-VLA (Zhao et al., 2025), Stamo (Liu et al., 2025b), Seer (Tian et al., 2025), DreamVLA (Zhang et al., 2025)) to simulate physical outcomes. Yet, without historical grounding, they risk hallucination under ambiguous observations. To address these limitations, ProRet-VLA unifies retrospection and prospection. Our Deviation-Driven Gating mechanism dynamically bridges retrieval and prediction, enhancing robustness in long-horizon tasks.

3 Method

In this section, we present **ProRet-VLA**. We first define the problem setup (Sec.3.1). Next, we introduce our retrieval-based episodic memory module for retrospection (Sec.3.2) and generative world model for prospection (Sec.3.3), followed by the

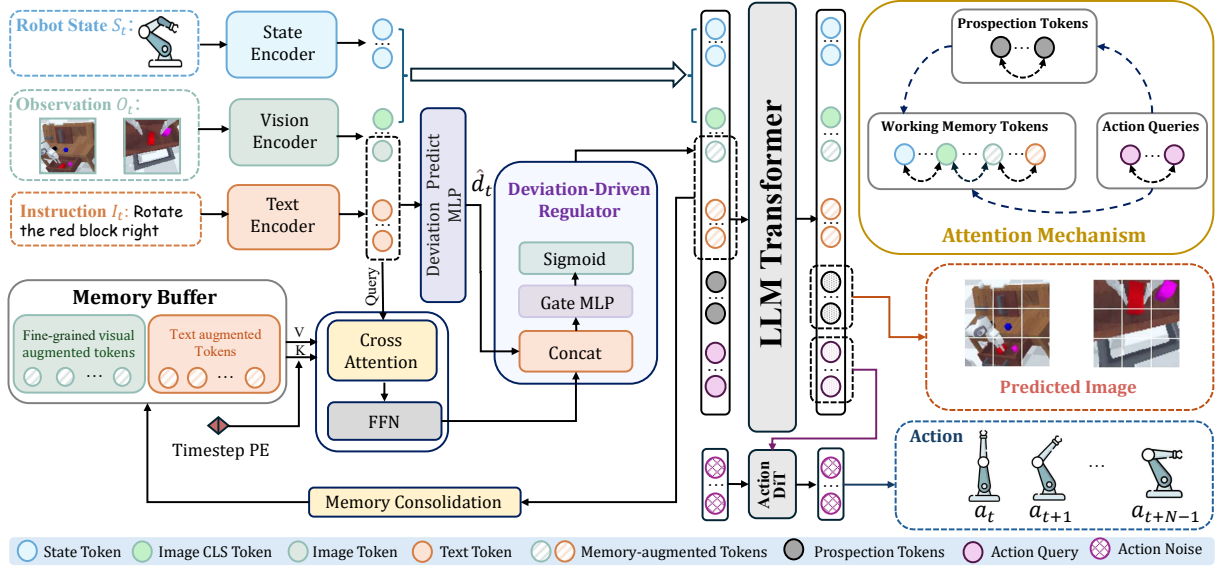


Figure 2: Overview of ProRet-VLA. Multimodal inputs (I_t, O_t, S_t) are encoded independently. Text tokens and a selected subset of visual tokens constitute the Current Context, which queries the Memory Buffer to retrieve and fuse historical context via adaptive gating. The Memory Buffer is updated through consolidation. Prospection Tokens (**P**) and Action Queries (**A**) are appended to the context; **P** predicts future latent states for foresight, while **A** generates the action chunk via a diffusion head.

Deviation-Driven Gating mechanism (Sec.3.4). Finally, we outline the diffusion-based action decoder and joint training objective (Sec.3.5) co-optimizing policy learning with temporal reasoning.

3.1 Preliminary

Problem Definition. The VLA paradigm aims to learn a policy π that maps a language instruction I_t , the current multi-view image observation O_t , and the robot’s proprioceptive state S_t to a continuous N -step action chunk $A_t = [a_t, a_{t+1}, \dots, a_{t+N-1}]$. Each action $a_t \in \mathbb{R}^7$ is a 7-DoF end-effector command $a_t = \{\Delta x, \Delta \theta, g\}$. Formally:

$$A_t = a_{t:t+N-1} = \pi(I_t, O_t, S_t), \quad (1)$$

where $\Delta x \in \mathbb{R}^3$ and $\Delta \theta \in \mathbb{R}^3$ denote translational and rotational offsets (e.g., in axis-angle or Euler), and $g \in \mathbb{R}$ is the gripper’s open-close state.

Our key insight is expanding the pre-action temporal receptive field. As shown in Fig. 2, a memory mechanism augments specific input tokens to accumulate historical experience for long-horizon tasks, then a set of learnable tokens **P** and **A** are appended to those multimodal tokens. A unified model $\mathcal{F}(\cdot)$ processes this sequence, decoding the tokens $\mathbf{h}_t^{\text{pred}}$ of **P** to predict future dynamics:

$$\mathbf{h}_t^{\text{pred}} = \mathcal{F}(\tilde{I}_t, \tilde{O}_t, S_t | \mathbf{P}), \quad (2)$$

and the output tokens of **A** condition a Diffusion Transformer $\mathcal{D}(\cdot)$ to generate the action chunk:

$$\hat{a}_{t:t+N-1} = \mathcal{D}(\mathcal{F}(\tilde{I}_t, \tilde{O}_t, S_t, \mathbf{P}) | \mathbf{A}). \quad (3)$$

Thus, we complete a Perception–Retrospection–Prospection–Action loop that is identical during training and inference.

Model Architecture. We leverage a large language model based on GPT-2 (Radford et al., 2019). We encode language instructions using CLIP (Radford et al., 2021) text embeddings, visual frames through a Masked Autoencoder (He et al., 2022), and proprioceptive signals via several convolutional and fully-connected layers.

3.2 Retrospection via Memory

We denote the Current Context as $\mathcal{C}_t = \{\mathbf{x}_1, \dots, \mathbf{x}_M\}$, which contains the multimodal tokens requiring temporal augmentation at t . These tokens query a Memory Buffer $\mathcal{B}_{t-1} = \{m_1, \dots, m_L\}$, where each m_i stores a compact representation of prior decision steps, and L denotes the buffer capacity. The buffer is updated within each episode, providing bounded historical memory without retaining raw observations.

Memory Retrieval. To preserve temporal order during retrieval, each buffered memory uses an episode-level temporal positional encoding $\text{PE}(\cdot)$ when constructing retrieval keys. We then stack the memories into tensors and apply scaled dot-product attention between the current tokens and the memory tensor to produce outputs:

$$K = [m_1 + \text{PE}(t_1); \dots; m_L + \text{PE}(t_L)], \quad (4)$$

$$\hat{R}_i = \text{softmax} \left(\frac{qK^T}{\sqrt{d}} \right) \mathcal{B}_{t-1}, \quad q = \mathbf{x}_i \in \mathcal{C}_t. \quad (5)$$

This attention is followed by a feed-forward network to complete one Transformer layer, yielding retrieved embeddings $\hat{\mathbf{R}}_t = [\hat{R}_1; \dots; \hat{R}_M]$.

In practice, we organize the Memory Buffer with modality-aware slots: fine-grained visual tokens and text tokens are retrieved separately to avoid mixing high-frequency perceptual changes with task-level semantics, while global visual and state tokens are kept as Direct Context; when the buffer exceeds its capacity, redundant adjacent entries are compressed to keep memory bounded. Additional details are provided in Appendix A.2.

3.3 Memory-Grounded Prospection

Forecasting future scene evolution from current observations alone is often under-constrained in long-horizon manipulation, especially under partial observability or delayed effects. We therefore introduce a memory-grounded prospection branch that predicts future observations from temporally enriched context. Rather than treating prediction merely as an auxiliary target, we use its predictive discrepancy to characterize prospective inconsistency, which provides the basis for deviation-driven memory fusion in Sec. 3.4.

Future Observation Prediction. To instantiate this branch, we introduce a set of learnable Prospection Tokens $\mathbf{P} \in \mathbb{R}^{K \times d}$. After memory regulation, these tokens are appended to the temporally enriched input sequence, which consists of the Direct Context and the Memory-Augmented Context. The Transformer backbone then maps the Prospection Tokens into hidden states $\mathbf{h}_t^{pro}$, allowing them to summarize the regulated temporal context for future prediction. To ensure these latents encode physically meaningful semantics, we employ a MAE decoder $\mathcal{D}(\cdot)$ to project them back into the pixel space. The prediction is generated as:

$$\hat{O}_{t+\delta} = \mathcal{D}(\mathbf{h}_t^{pred}). \quad (6)$$

During training, this generative world model is optimized via a reconstruction objective:

$$\mathcal{L}_{pro} = \|\hat{O}_{t+\delta} - O_{t+\delta}\|^2. \quad (7)$$

This self-supervised task encourages the model to learn predictive visual dynamics and provides the basis for estimating prospective inconsistency. During inference, we skip the decoder entirely to save

computation, as it is only used to supervise future prediction during training.

3.4 Deviation-Driven Gating Mechanism

The retrieved embeddings $\hat{\mathbf{R}}_t$ provide retrospective memory but should not be fused with the Current Context indiscriminately. Their usefulness varies across decision stages: they may be redundant under clear observations yet crucial under ambiguity, occlusion, or unexpected dynamics. Therefore, the key question is not whether to use memory, but how to regulate its contribution according to the agent’s current predictive state.

Deviation Signal. We use the prediction error of the prospection branch as a training-time measure of prospective inconsistency. Given the predicted future observation $\hat{O}_{t+\delta}$ and the observed future frame $O_{t+\delta}$, we define the deviation target as:

$$d_t = \text{sg} \left[\tanh \left(\left\| \hat{O}_{t+\delta} - O_{t+\delta} \right\|_2 \right) \right], \quad (8)$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator, which preserves its input during the forward pass but blocks gradients during backpropagation. Therefore, d_t is recomputed at each training iteration but is treated as a fixed pseudo-label during the corresponding backward pass.

A larger d_t indicates that the current temporal context is less predictive of the future evolution. We emphasize that this deviation target is not intended to be a calibrated failure probability or a complete measure of physical uncertainty. Since d_t is unavailable during real-time inference, we employ a lightweight Deviation Predictor $\mathcal{P}(\cdot)$ to estimate it from the Current Context: $\mathcal{C}_t$:

$$\hat{d}_t = \mathcal{P}(\mathcal{C}_t) \in [0, 1]. \quad (9)$$

This predictor is trained via an auxiliary loss :

$$\mathcal{L}_{dev} = \|\hat{d}_t - d_t\|^2. \quad (10)$$

Thus, the gate does not access future observations at test time; it only uses a predicted deviation learned from self-supervised future consistency.

Memory Fusion. Finally, we use the predicted deviation to condition memory fusion. For each current token $\mathbf{x}_i$ and its retrieved counterpart $\hat{R}_i$, the fusion coefficient is computed by a Gate MLP G_θ conditioned on the current feature, the retrieved memory, and the predicted deviation:

$$\lambda_i = \sigma(G_\theta[\mathbf{x}_i; \hat{R}_i; \hat{d}_t]). \quad (11)$$

Table 1: CALVIN ABC-D results. The **Guidance** column categorizes methods based on their temporal reasoning mechanism: using memory module (*Mem*), predictive world models (*Pred*), both (*Both*), or neither (-). Columns 1–5 represent the success rates (%) of completing the number of instructions in a row, computed over 1000 rollouts. The last column reports the average number of completed tasks (Avg. Len.). ProRet-VLA shows improvements over baselines. The best results are **bolded**.

Method	Guidance	1	2	3	4	5	Avg. Len. ↑
GR-1 (Wu et al., 2024)	-	85.4	71.2	59.6	49.7	40.1	3.06
π_0 (Black et al., 2024)	-	93.8	85.0	76.7	68.1	59.9	3.92
OpenVLA-OFT (Kim et al., 2025)	-	96.3	89.1	82.4	75.8	66.5	4.10
CLOVER (Yang, 2024)	Pred	96.0	83.5	70.8	57.5	45.4	3.53
GR-MG (Li et al., 2025a)	Pred	96.8	89.3	81.5	72.7	64.4	4.04
Seer-Large (Tian et al., 2025)	Pred	96.3	91.6	86.1	80.3	74.0	4.28
VPP (Hu et al., 2024)	Pred	95.7	91.2	86.3	81.0	75.0	4.29
DreamVLA (Zhang et al., 2025)	Pred	98.2	94.6	89.5	83.4	78.1	4.44
RoboFlamingo (Li et al., 2024c)	Mem	82.4	61.9	46.6	33.1	23.5	2.47
UniVLA (Bu et al., 2025)	Mem	95.5	85.8	75.4	66.9	56.5	3.80
RoboVLMs (Li et al., 2024b)	Mem	98.0	93.6	85.4	77.8	70.4	4.25
ProRet-VLA (Ours)	Both	98.2	94.9	90.3	86.3	79.9	4.50

The σ denotes the sigmoid activation and final Memory-Augmented Context is obtained as:

$$\tilde{\mathbf{x}}_i = \lambda_i \mathbf{x}_i + (1 - \lambda_i) \hat{R}_i. \quad (12)$$

Here, $\hat{d}_t$ is not a hand-crafted monotonic bias, but a condition that lets the gate learn how predictive deviation should affect memory fusion. As shown in Sec.4.4, the learned gate is empirically inversely correlated with predictive deviation, indicating a shift toward retrospective memory during high-deviation phases.

3.5 Action Decoding with Temporal Context

The final objective is to infer the optimal action sequence $\hat{a}_{t:t+N-1}$. To achieve this, we append a set of learnable Action Queries $\mathbf{A} \in \mathbb{R}^{N \times d}$ where N denotes the action chunk size, after the Prospection Tokens. Under the structured attention mask, these queries can attend to the Direct Context, the deviation-regulated Memory-Augmented Context, and the Prospection Tokens. The resulting action hidden states therefore summarize the information needed for continuous control. These hidden states are then used as conditioning features for a diffusion Transformer action decoder, whose architectural details are provided in Appendix A.3.

Training Objective. Our model is trained in an end-to-end manner, jointly optimizing for action generation, future simulation, and deviation estimation. The total training objective is a weighted sum of three components:

$$\mathcal{L}_{total} = \lambda_{DiT} \mathcal{L}_{DiT} + \lambda_{pro} \mathcal{L}_{pro} + \lambda_{dev} \mathcal{L}_{dev}. \quad (13)$$

Here, $\mathcal{L}_{pro}$ trains the prospection branch, $\mathcal{L}_{dev}$ supervises the deviation predictor, and $\mathcal{L}_{DiT}$ optimizes the final control policy. By co-optimizing

these objectives, the gradients from the auxiliary tasks back-propagate into the backbone.

4 Experiments

We first introduce our experimental setup (Sec. 4.1), followed by a comparison with state-of-the-art baselines on benchmarks (Sec. 4.2). Next, we conduct ablation studies to isolate the contributions of prospection, retrospection, and deviation-conditioned gating (Sec. 4.3). We further analyze the gating behavior through qualitative visualization and high-deviation event analysis (Sec. 4.4). Finally, we evaluate ProRet-VLA on real-world long-horizon manipulation tasks (Sec. 4.5).

4.1 Experimental Setup

Simulation Benchmarks. We evaluate our method on two widely recognized robotic manipulation benchmarks for long-horizon tasks. These environments pose challenges due to their requirement for extended temporal reasoning:

- **CALVIN ABC-D** (Mees et al., 2022): This benchmark includes four indoor environments (A-D); policies are trained on A-C and evaluated on the unseen D to assess generalization on consecutive tasks.
- **LIBERO** (Liu et al., 2023): This benchmark comprises four distinct task suites: LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long, with LIBERO-Long posing significant challenges for long-horizon tasks.

All experiments are conducted under the setting following: a multiview setup using both the primary and wrist cameras.

Table 2: LIBERO results. We report the Success Rate (SR, %) and Rank for each method across all tasks. Each task is evaluated over 500 trials. The **Guidance** column categorizes methods based on their temporal reasoning mechanism: using memory module (*Mem*), predictive world models (*Pred*), both (*Both*), or neither (-). ProRet-VLA achieves superior average performance.

Method	Guidance	Spatial		Object		Goal		Long		Average	
		SR	Rank	SR	Rank	SR	Rank	SR	Rank	SR	Rank
Diffusion Policy (Chi et al., 2025)	-	78.3	11	92.5	7	68.3	11	50.5	11	72.4	11
OpenVLA (Kim et al., 2024)	-	84.7	8	88.4	9	79.2	9	53.7	9	76.5	9
SpatialVLA (Wu et al., 2024)	-	88.2	7	89.9	8	78.6	10	55.5	8	78.1	8
CogACT (Li et al., 2024a)	-	97.2	4	98.0	4	90.2	6	88.8	5	93.2	5
π_0 (Black et al., 2024)	-	96.8	5	98.8	1	95.8	3	85.2	7	94.2	4
CoT-VLA (Zhao et al., 2025)	Pred	81.1	9	87.5	10	91.6	5	87.6	6	83.4	7
DreamVLA (Zhang et al., 2025)	Pred	97.5	3	94.0	6	89.5	7	89.5	4	92.6	6
Octo (Ghosh et al., 2024)	Mem	78.9	10	85.7	11	84.6	8	51.1	10	75.1	10
MemoryVLA (Shi et al., 2025)	Mem	98.4	1	98.4	3	96.4	2	93.4	2	96.5	2
UniVLA (Bu et al., 2025)	Mem	96.5	6	96.8	5	95.6	4	92.0	3	95.2	3
ProRet-VLA (Ours)	Both	97.6	2	98.5	2	96.7	1	93.9	1	96.7	1

Implementation Details. All models are implemented in PyTorch and trained on 8 NVIDIA L20 GPUs. We use an AdamW (Loshchilov and Hutter, 2017) optimizer with initial learning rate 10^{-3} , weight decay $1e-4$, and a cosine learning rate schedule with 5% linear warm-up. Batch size is set to 64, we set the query length of prospection to 9 and diffusion steps in DiT to 10, we set the memory length to $L = 16$ based on controlled memory ablations in Appendix B.4. We weight the prospection loss as $\lambda_{pro} = 1$, the deviation loss as $\lambda_{dev} = 0.1$ and the action loss as $\lambda_{DiT} = 1$, respectively. We first pretrain ProRet-VLA on the language-free split of the CALVIN and on the full DROID dataset (Khazatsky et al., 2024). For the LIBERO benchmark, we pretrain ProRet-VLA on LIBERO-90. Fine-tuning is conducted for 40 epochs on LIBERO and 20 epochs on CALVIN. Additional details are provided in Appendix A.4.

4.2 Main Results

Experimental Results on CALVIN. As shown in Tab. 1, ProRet-VLA achieves an average sequence length of 4.50, the highest among the compared methods. It ties DreamVLA at the one-instruction horizon, with both methods achieving a success rate of 98.2%, and obtains the highest success rates at horizons of two to five instructions.

The performance difference is more pronounced at longer horizons. For five consecutive instructions, ProRet-VLA achieves a success rate of 79.9%, exceeding DreamVLA at 78.1% and VPP at 75.0% by 1.8 and 4.9 percentage points, respectively. These results show that the gains of ProRet-VLA are concentrated in multi-step task completion rather than single-task execution

Experimental Results on LIBERO. Tab. 2 summarizes the success rates across the four distinct task suites. ProRet-VLA achieves highly competitive performance, delivering an average success rate of 96.7%, which is comparable to or surpasses the strong baselines presented in this evaluation. Specifically, compared to MemoryVLA at 96.5% and UniVLA at 95.2%, our method demonstrates a consistent advantage, particularly in the LIBERO-Goal suite where it attains 96.7%. While π_0 shows slightly superior performance in single-object manipulation with 98.8%, ProRet-VLA effectively bridges the gap in complex, long-horizon scenarios in LIBERO-Long, outperforming π_0 by 8.7 percentage points. This indicates that our framework offers a balanced and robust solution across diverse manipulation challenges.

4.3 Ablation Studies

To systematically verify the distinct impact of individual components, we conducted ablation studies on the CALVIN ABC-D benchmark.

Ablation on Components. We first isolate the effects of prospection and retrospection on CALVIN ABC-D. All variants are trained with the same data, optimizer, schedule, backbone, and action decoder; only the temporal module is changed. Here, *feature-only gated fusion* denotes a learned token-wise gate conditioned only on the current token and its retrieved memory, i.e., without the predicted deviation input $\hat{d}_t$ used in ProRet-VLA. Detailed definitions of the ablation variants are provided in Appendix B.3. As shown in Table 3, the vanilla baseline achieves an average sequence length of 3.64. Adding retrospective retrieval with feature-only gated fusion (w/ Ret) improves the result to

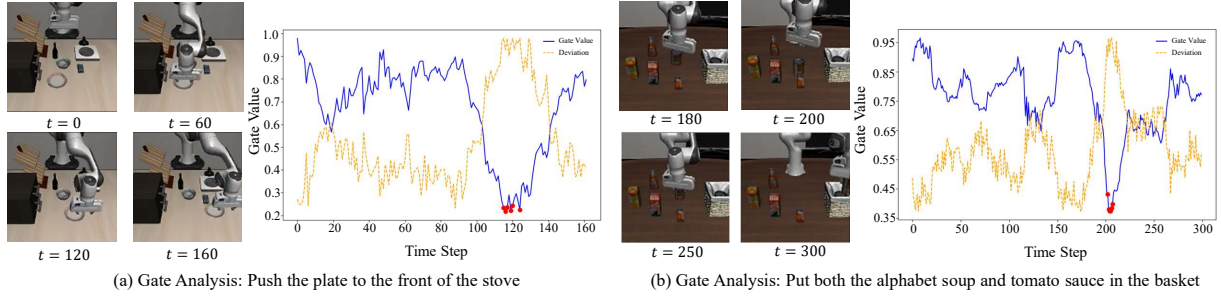


Figure 3: Visualization of Temporal Gating Behaviors. The plots illustrate the dynamic evolution of the Gate Value (blue solid line) and Prediction Deviation (yellow dashed line) during the execution of two tasks: (a) Push the plate to the front of the stove and; (b) Put both the alphabet soup and tomato sauce in the basket.

Table 3: Ablation study of component contributions and gating strategies. We compare ours with the vanilla baseline, single-component variants, fixed fusion ratios, feature-only learned gating, and deviation-intervention variants.

Methods	1	2	3	4	5	Avg. Len.↑
Baseline	93.0	82.4	72.3	62.6	53.3	3.64
w/ Ret	94.4	88.5	82.7	77.3	68.9	4.12
w/ Pro	96.3	91.6	86.1	80.3	74.0	4.28
$\lambda = 0.2$	93.9	88.1	81.9	77.1	68.3	4.09
$\lambda = 0.5$	95.3	90.1	84.9	79.8	74.8	4.22
$\lambda = 0.8$	97.5	92.7	87.2	80.8	74.2	4.32
Learned Gate	98.5	93.7	87.2	81.5	74.8	4.38
Constant Deviation	97.1	92.4	86.8	80.1	73.9	4.30
Shuffled Deviation	93.7	86.7	79.9	72.8	65.8	4.00
Ours	98.2	94.9	90.3	86.3	79.9	4.50

4.12, while using only the prospection branch to predict future observations from the current context (w/ Pro) further increases it to 4.28. However, both single-component variants remain below the full ProRet-VLA, which reaches 4.50. These results indicate that historical memory and future prediction provide complementary temporal cues, and that coupling them through deviation-conditioned memory fusion is more effective than using either component alone.

Ablation on Gating Strategies. We next examine how current context and retrieved memory should be fused. As shown in Table 3, increasing the fixed gate ratio from $\lambda = 0.2$ to 0.8 improves the average length from 4.09 to 4.32, suggesting that current observations should remain a stable anchor for control. However, all fixed ratios underperform ProRet-VLA, indicating that a static mixture cannot adapt across decision stages. The feature-only *Learned Gate* reaches an average length of 4.38, but still trails our model by 0.12 in average length and 5.1 percentage points in 5-step success, showing that learnable token-level fusion alone is insufficient. Deviation interventions further confirm this: replacing $\hat{d}_t$ with a constant

value of 0.5 reduces the average length to 4.30, while shuffling cached deviations across trajectories or timesteps drops it to 4.00. Since shuffling preserves the marginal distribution of deviations but breaks their alignment with the current decision state, the gain comes from temporally aligned predictive deviation rather than from injecting an arbitrary scalar signal.

4.4 Analysis of Deviation-Driven Gating

Visualization of Gating Behaviors. To analyze the Deviation-Driven Gating mechanism, Figure 3 visualizes the temporal dynamics of gate values versus prediction deviations across representative LIBERO tasks. We find that under stable conditions, high gate values indicate reliance on the current observation. Conversely, during critical high-deviation phases (marked by red dots, e.g., $t \approx 120$ in (a) and $t \approx 200$ in (b)), the gate value drops significantly. This triggers a cognitive shift, enabling the agent to handle complex situations.

High-deviation event analysis. Beyond qualitative visualization, we further quantify whether deviation-driven gating helps policies maintain task progress after temporally uncertain phases. Since some variants do not contain a Deviation Predictor, we use the Deviation Predictor trained in ProRet-VLA as a frozen monitor for all methods during evaluation. We further analyze whether the predicted deviation aligns with the self-supervised deviation target in Appendix B.6.

This monitor is used only to detect high-deviation events and does not affect the action generation of the baselines. For each variant, τ_{70} denotes the 70th percentile of the monitored deviation values, and a high-deviation event is detected when the deviation exceeds this threshold. *Evt.* reports the frequency of detected events. *Comp.* is computed as the ratio of high-deviation events after

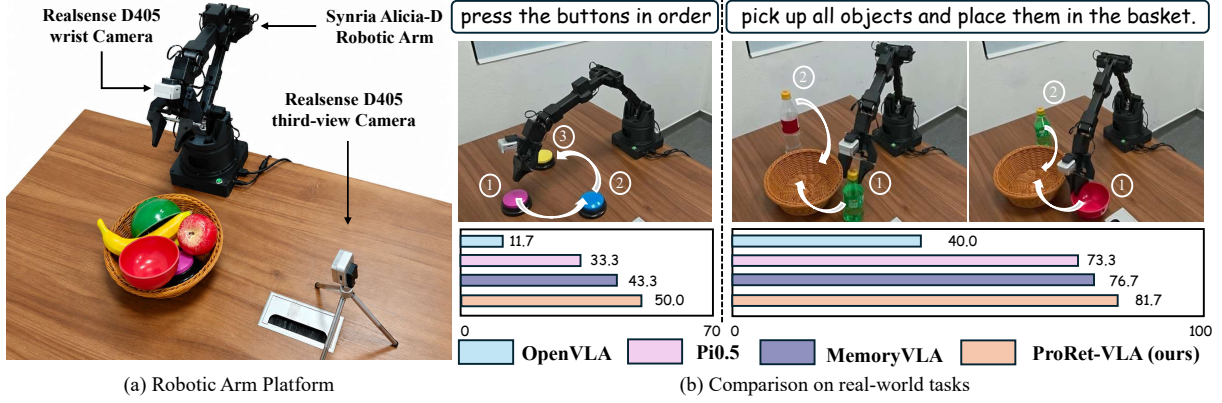


Figure 4: Real-world long-horizon tasks. (a) We deploy our system on the Synria Alicia-D robot arm equipped with two Intel RealSense D405 cameras: a fixed third-person scene camera and a wrist-mounted camera. (b) Performance comparison on real world experiments, Success scores (%) are reported over two long-horizon temporal tasks.

Table 4: High-deviation event analysis on CALVIN ABC-D. High-deviation events are detected using the 70th percentile threshold τ_{70} . Comp. measures post-event subtask completion, and Post Len. measures task progress after the first high-deviation event.

Method	τ_{70}	Evt.	Comp. $\uparrow$	Post Len. $\uparrow$
w/ Pro	0.84	0.40	0.88	3.23
Fixed $\lambda = 0.8$	0.83	0.47	0.83	3.26
Learned Gate	0.88	0.43	0.91	3.43
Ours	0.92	0.43	0.93	3.48

which the current subtask is completed within the next 100 environment steps. *Post Len.* measures how many tasks the policy can still complete after the first high-deviation event in a rollout.

As shown in Tab. 4, ProRet-VLA achieves the highest *Comp.* of 0.93 and the best *Post Len.* of 3.48. Compared with the fixed-gate variant, our method improves *Comp.* from 0.83 to 0.93, showing that static fusion is less effective at maintaining task progress after high-deviation phases. The feature-only learned gate achieves the same *Evt.* as ours, but lower *Comp.* and *Post Len.*, indicating that the gain does not come from fewer detected high-deviation events. Instead, temporally aligned predictive deviation helps ProRet-VLA regulate memory fusion and achieve stronger post-event task continuation after high-deviation events.

4.5 Real World Experiments

Experiment setups. To evaluate the effectiveness of our approach in real-world applications, we conduct real-world experiments using the Synria Alicia-D robot. As shown in Fig. 4, the system uses two Intel RealSense D405 cameras: one fixed third-person camera for scene-level observation and one wrist-mounted camera for egocentric observation. We collect two long-horizon tasks, each with 100

demonstrations and train each model separately for every task

Real-World Evaluation. For each model and task, we report the mean success rate over three independent evaluation runs of 20 trials each, resulting in 60 trials per model–task pair. Press Buttons in Order has a direct memory dependency because the policy must track previously completed subgoals in the specified sequence. Pick All Objects primarily evaluates temporal progress tracking and additionally involves challenging object grasping. In particular, grasping the bowl is the dominant failure mode because selecting a stable contact point along its rim remains difficult.

As shown in Fig. 4(b), OpenVLA achieves success rates of 11.7% and 40.0% on Press Buttons in Order and Pick All Objects, respectively, while $\pi_{0.5}$ achieves 33.3% and 73.3%. MemoryVLA obtains 43.3% and 76.7%, whereas ProRet-VLA achieves the highest observed success rates of 50.0% and 81.7%. These results are 6.7 and 5.0 percentage points higher than MemoryVLA under the matched data and evaluation protocol. This improvement is mainly because ProRet-VLA can leverage retrospective temporal context to recognize completed subgoals, such as pressed buttons or already placed objects, thereby maintaining more consistent long-horizon execution in real-world environments. Real-world examples for both tasks are provided in Appendix B.8.

5 Conclusion

We presented ProRet-VLA, a temporal reasoning framework that connects prospective future prediction with retrospective episodic memory through predictive consistency. ProRet-VLA uses future

prediction error to supervise a Deviation Predictor, whose output conditions token-wise memory fusion during inference. Experiments on CALVIN ABC-D, LIBERO, and real-world tasks show improved long-horizon task completion and temporal progress tracking over strong VLA baselines.

Limitations

ProRet-VLA still has several limitations. First, although the memory buffer stores compact latent representations rather than raw observations, these latents remain relatively high-dimensional and may contain redundant or stale information in very long episodes. Second, the Deviation Predictor is supervised by future prediction error and estimates prospective inconsistency only from the current context during inference; therefore, its calibration may degrade under large distribution shifts, unseen failure modes, or substantially different robot platforms. Third, our real-world evaluation covers two long-horizon manipulation tasks with limited variations, which validates practical feasibility but does not fully characterize performance in more diverse real-world settings such as dynamic scenes, heavy clutter, deformable objects, or human-robot interaction. Fourth, our simulation benchmark results follow the standard large-rollout evaluation protocol, but we do not report multi-seed training variance due to computational constraints, which limits the statistical characterization of close comparisons. Future work will explore sparser and more task-adaptive memory representations, better-calibrated deviation estimation, repeated training runs or bootstrap confidence intervals, and broader real-world evaluations.

Acknowledgment

This research is supported by the National Natural Science Foundation of China (No.62476127,62306140), the Natural Science Foundation of Jiangsu Province (No.BK20242039), the Fundamental Research Fund for the Central Universities (NO.NXD2026006), the Disciplinary Basic Research Project Fund (No.ILF26023), the Research Fund (No.PO250624101698), the Scientific Research Starting Foundation of Nanjing University of Aeronautics and Astronautics (No.YQR21022), and the High Performance Computing Platform of Nanjing University of Aeronautics and Astronautics.

References

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, and 1 others. 2022. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*.
- Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, and 1 others. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.
- Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, and 1 others. 2024. π_0 : A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*.
- Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. 2022. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*.
- Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. 2025. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*.
- Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. 2025. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*.
- Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. 2021. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*.
- Karl Friston. 2010. The free-energy principle: a unified brain theory? *Nature reviews neuroscience*.
- Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, and 1 others. 2024. Octo: An open-source generalist robot policy. In *Robotics: Science and Systems*.
- Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

- Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. 2024. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*.
- Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, and 1 others. 2025. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*.
- Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. 2024. Prismatic vlms: investigating the design space of visually-conditioned language models. In *Proceedings of the 41st International Conference on Machine Learning*.
- Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, and 1 others. 2024. Droid: A large-scale in-the-wild robot manipulation dataset. In *RSS 2024 Workshop: Data Generation for Robotics*.
- Moo Jin Kim, Chelsea Finn, and Percy Liang. 2025. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*.
- Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, and 1 others. 2024. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*.
- Myungkyu Koo, Daewon Choi, Taeyoung Kim, Kyungmin Lee, Changyeon Kim, Younggyo Seo, and Jinwoo Shin. 2025. Hamlet: Switch your vision-language-action model into a history-aware policy. *arXiv preprint arXiv:2510.00695*.
- Peiyan Li, Hongtao Wu, Yan Huang, Chilam Cheang, Liang Wang, and Tao Kong. 2025a. Gr-mg: Leveraging partially-annotated data via multi-modal goal-conditioned policy. *IEEE Robotics and Automation Letters*.
- Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, and 1 others. 2024a. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *CoRR*.
- Runhao Li, Wenkai Guo, Zhenyu Wu, Changyuan Wang, Haoyuan Deng, Zhenyu Weng, Yap-Peng Tan, and Ziwei Wang. 2025b. Map-vla: Memory-augmented prompting for vision-language-action model in robotic manipulation. *arXiv preprint arXiv:2511.09516*.
- Xinghang Li, Peiyan Li, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Tao Kong, Hanbo Zhang, and Huaping Liu. 2024b. Towards generalist robot policies: What matters in building vision-language-action models. *CoRR*.
- Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, and 1 others. 2024c. Vision-language foundation models as effective robot imitators. In *ICLR*.
- Minghui Lin, Pengxiang Ding, Shu Wang, Zifeng Zhuang, Yang Liu, Xinyang Tong, Wenxuan Song, Shangke Lyu, Siteng Huang, and Donglin Wang. 2025. Hif-vla: Hindsight, insight and foresight through motion representation for vision-language-action models. *arXiv preprint arXiv:2512.09928*.
- Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. 2023. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*.
- Jiaming Liu, Hao Chen, Pengju An, Zhuoyang Liu, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, and 1 others. 2025a. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*.
- Mingyu Liu, Jiuhe Shu, Hui Chen, Zeju Li, Canyu Zhao, Jiange Yang, Shenyuan Gao, Hao Chen, and Chunhua Shen. 2025b. Stamo: Unsupervised learning of generalizable robot motion from compact state representation. *arXiv preprint arXiv:2510.05057*.
- Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.
- Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. 2022. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*.
- Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandekar, Ajinkya Jain, and 1 others. 2024. 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)*.
- William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models

- from natural language supervision. In *International conference on machine learning*.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.
- Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. 2025. Memoryvla: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. *arXiv preprint arXiv:2508.19236*.
- Ajay Sridhar, Jennifer Pan, Satvik Sharma, and Chelsea Finn. 2025. Memer: Scaling up memory for robot control via experience retrieval. *arXiv preprint arXiv:2510.20328*.
- Yang Tian, Sizhe Yang, Jia Zeng, Ping Wang, Dahua Lin, Hao Dong, and Jiangmiao Pang. 2025. Predictive inverse dynamics models are scalable learners for robotic manipulation. In *The Thirteenth International Conference on Learning Representations*.
- Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. 2024. Unleashing large-scale video generative pre-training for visual robot manipulation. In *ICLR*.
- Yanchao Yang. 2024. Closed-loop visuomotor control with generative expectation for robotic manipulation. In *The Conference on Neural Information Processing Systems (10/12/2024-15/12/2024, Vancouver)*.
- Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, XinQiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, He Wang, Zhizheng Zhang, and 1 others. 2025. Dreamvla: A vision-language-action model dreamed with comprehensive world knowledge. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, and 1 others. 2025. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*.
- Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. 2024. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *arXiv preprint arXiv:2412.10345*.
- Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, and 1 others. 2023. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*.

A Implementation Details

A.1 ProRet-VLA Architecture

Robot State Encoder. The robot state consists of the arm and gripper state. The arm state includes the end-effector position and its rotation in Euler angles, resulting in a six-dimensional representation. The gripper state is a binary value indicating whether the gripper is open or closed. We tokenize the robot state using an MLP. Specifically, the gripper state is first converted into a one-hot encoding. The one-hot encoding of the gripper state and the arm state are then each passed through separate linear layers. The outputs are concatenated and passed through a final linear layer to produce the state token.

Text Encoder. We use the CLIP ViT-B/32 text encoder (Radford et al., 2021) to process natural language task instructions. The encoder transforms each instruction into a fixed-length embedding that captures semantic intent. These embeddings are then projected into the shared latent space and used to condition the subsequent modules, enabling effective grounding of language into perception and action.

Visual Encoder. We employ an MAE-pretrained ViT-B (He et al., 2022) as the vision encoder. At each timestep, images are captured from two views: eye-on-hand and eye-on-base. Each image is processed by the vision encoder to produce 196 latent vectors, which represent local patch information, along with a CLS token that encodes the global representation of the image. Directly inputting all 197 tokens into the transformer backbone would create a significant computational burden, particularly when processing long histories. Moreover, many image details are redundant for accomplishing manipulation tasks. To address this, we utilize the Perceiver Resampler (Alayrac et al., 2022) to condense the image representations and extract task-relevant features. The Perceiver Resampler employs learnable latent vectors with a shape of (num latents, dim), where num latents is significantly smaller than the number of image tokens. Through Perceiver Attention, these latent vectors condense the input image features, along with the CLS token, to form the final image tokens.

Learnable Queries. Following encoding, a set of learnable queries **P** (Prospection Tokens) and **A** (Action Queries) are appended to these multi-

Table 5: Hyperparameters for the transformers in our policy.

Module	Hidden size	Layers	Heads
Image encoder	768	12	12
Perceiver resampler	768	3	8
Transformer backbone	1024	24	16
Image decoder	1024	2	16

modal embeddings to read out foresight and actions. Specifically, **P** are appended to extract representations for two views, and three **A** are appended to predict actions across three steps, ensuring temporal action consistency and robustness to idle actions.

Decoder. For image decoding, following (He et al., 2022), we use a vision transformer (ViT) as the image decoder. The image decoder receives the image latent and mask tokens from the transformer backbone as input. Positional information is provided through fixed sine-cosine positional encodings. The inputs are processed by multiple transformer encoder blocks. Finally, a linear layer predicts the pixels for each patch, generating the image that represents the predicted future state. Concrete details of each module are shown in Table 5.

Large Language Model. We adopt GPT-2 Medium (Radford et al., 2019) as our language backbone. GPT-2 Medium is a 24-layer, 16-head Transformer decoder with a hidden size of 1024 and a total of approximately 345 million parameters. It was pretrained on the WebText corpus (~ 8 million documents, 40 GB of text) using autoregressive language modeling to predict the next token with a byte-pair encoding vocabulary of 50257 tokens.

Deviation Predictor. To estimate the potential prediction error during inference, we implement the Deviation Predictor as a lightweight Multi-Layer Perceptron (MLP). It takes the concatenation of the current instruction representation and fine-grained visual tokens as input. The network first projects the concatenated feature into a hidden dimension of 128, followed by Layer Normalization and a GELU activation. A final linear layer maps the feature to a scalar, which is then passed through a sigmoid activation to produce the predicted deviation signal $\hat{d}_t \in [0, 1]$.

A.2 Memory Module.

Dual-Stream Memory Architecture. Adopting the dual-stream architectural design from Memo-

ryVLA (Shi et al., 2025), we design a dual-layer memory architecture to decouple low-level spatiotemporal details from high-level semantic abstractions. We explicitly bifurcate the MAE visual output into local patch embeddings $\mathbf{E}_t \in \mathbb{R}^{N \times d}$ and a global class token $\mathbf{c}_t \in \mathbb{R}^{1 \times d}$. The former preserves fine-grained geometric details for spatiotemporal retrieval while the latter encapsulates the global scene gist. Together with the semantic tokens $\mathbf{L}_t$ and state embeddings $\mathbf{S}_t$, we devise a bifurcated processing strategy driven by the necessity to balance stable physical grounding with long-term reasoning.

We designate $\mathbf{c}_t$ and $\mathbf{S}_t$ as *Direct Context*. These tokens bypass memory retrieval and serve as a drift-free anchor to the immediate physical status. In contrast, $\mathbf{E}_t$ and $\mathbf{L}_t$ constitute the *Memory-Augmented Context*. They act as queries to retrieve historical experiences from two decoupled streams: a *Perceptual Stream* where visual patches query $\mathcal{B}_{per}$ and a *Cognitive Stream* where text queries $\mathcal{B}_{cog}$, that is, we establish two Memory Buffers to separately preserve different tokens. This separation prevents interference between high-frequency visual dynamics and low-frequency semantic planning, enabling the agent to simultaneously recall physical changes and logical sub-task progression.

Memory Consolidation. We use ToMe-inspired (Bolya et al., 2022) adjacent-entry consolidation when the buffer exceeds its capacity. The adjacent pair with the highest cosine similarity is merged by averaging their feature tensors, keeping the memory size bounded throughout long-horizon execution.

Implementation. In our implementation, both memory streams operate in the shared backbone latent space with hidden dimension $d = 1024$. Each stream maintains an episodic buffer with a maximum capacity of $L = 16$ entries. At each timestep, tokens in the Memory-Augmented Context retrieve historical information from their corresponding buffer through two stacked cross-attention–FFN retrieval blocks. To preserve temporal order, each stored entry is augmented with a timestep embedding computed from sinusoidal frequency features before retrieval. For memory fusion, we use a deviation-conditioned gate that takes the current token, the retrieved memory token, and the predicted deviation $\hat{d}_t$ as inputs. The deviation predictor is supervised by the future prediction error of the prospection branch, computed from the difference

between the predicted future observation $\hat{O}_{t+\delta}$ and the observed future frame $O_{t+\delta}$. During training, the future frame is used only as a self-supervised target for the prospection and deviation losses. During inference, future observations are unavailable, and the gate relies solely on the predicted deviation estimated from the current context.

A.3 Action Prediction with Diffusion Transformer.

For continuous action generation, we adopt the diffusion-based Transformer decoder from DreamVLA, instantiated with the DiT-B architecture (Peebles and Xie, 2023). After the VLA backbone processes the direct context, deviation-regulated memory context, and prospection tokens, the output hidden states of the Action Queries are used as conditioning features for the diffusion action decoder.

The action decoder operates in the same latent space as the backbone, with the conditioning feature dimension set to $d = 1024$. It predicts a 3-step action chunk, where each action is a 7-dimensional end-effector command consisting of 6-DoF pose displacement and a gripper state. In the DiT implementation, this corresponds to an action prediction length of 3, with a future action window size of 2 and no past action context.

During training, Gaussian noise is added to the ground-truth action trajectories, and the DiT decoder is optimized to denoise them conditioned on the Action Query features. During inference, actions are sampled from Gaussian noise through deterministic diffusion denoising with 10 sampling steps. This diffusion formulation allows the policy to model multimodal continuous action distributions while preserving temporal consistency across the predicted action chunk.

A.4 Training Detail

The total loss can be formulated as:

$$\mathcal{L}_{total} = \lambda_{DiT} \mathcal{L}_{DiT} + \lambda_{pro} \mathcal{L}_{pro} + \lambda_{dev} \mathcal{L}_{dev}. \quad (14)$$

We weight the prospection loss as $\lambda_{pro} = 1$, the deviation loss as $\lambda_{dev} = 0.1$ and the action loss as $\lambda_{DiT} = 1$, respectively. All models are implemented in PyTorch and trained on NVIDIA 8 L20 GPUs. We use an AdamW (Loshchilov and Hutter, 2017) optimizer with initial learning rate 10^{-3} , weight decay $1e - 4$, and a cosine learning rate schedule with 5% linear warm-up. Batch size is set

Table 6: Ablation study of memory hyperparameters on CALVIN ABC-D. We evaluate the effects of buffer type, buffer length, and consolidation strategy under the same ProRet-VLA architecture and training protocol. Columns 1–5 report the success rates (%) of completing the corresponding number of consecutive instructions, and Avg. Len. denotes the average number of completed tasks. The best configuration in each group is highlighted.

Setting	Variant	1	2	3	4	5	Avg. Len. $\uparrow$
Buffer Type	Unified buffer	98.1	93.9	88.8	82.9	77.7	4.41
	Dual-stream buffer	98.2	94.9	90.3	86.3	79.9	4.50
Buffer Length	$L = 8$	97.1	92.6	89.0	82.5	76.3	4.38
	$L = 16$	98.2	94.9	90.3	86.3	79.9	4.50
	$L = 32$	98.2	94.6	90.1	84.7	78.6	4.46
Consolidation Type	FIFO	97.4	93.1	88.4	82.1	75.9	4.37
	ToMe	98.2	94.9	90.3	86.3	79.9	4.50

to 64, we set the query length of prospection to 9 and diffusion steps in DiT to 10, we also set length of Memory Bank to 16.

During training, we freeze the CLIP text encoder and the MAE vision encoder to preserve their pre-trained representations and reduce the optimization cost. All remaining task-specific modules are trained end-to-end, including the Perceiver resampler, Transformer backbone, memory retrieval and consolidation modules, deviation-conditioned gate, Deviation Predictor, image/prospection decoder, and DiT action decoder. The total computational budget is approximately 730–750 GPU-hours, estimated on 8 NVIDIA L20 GPUs.

All public benchmarks and pretrained components are used for research purposes following their released terms and licenses. Our real-world demonstrations are collected in a controlled laboratory setting and are used only for research evaluation.

B Experimental Results

B.1 Evaluation protocol and baseline comparison.

For public benchmark comparisons, we follow the standard evaluation protocols of CALVIN ABC-D and LIBERO. CALVIN results are reported over 1000 rollouts, and each LIBERO task is evaluated over 500 trials. Since many compared methods use different pretrained backbones, pretraining data, and action decoders, the main tables should be interpreted as benchmark-level comparisons rather than strictly compute-matched comparisons. To isolate the effect of our design, all internal ablations in Sec. 4.3 are trained and evaluated under the same data, optimizer, schedule, backbone, action decoder, and evaluation protocol, with only the temporal reasoning module changed. All simulation results are evaluated under the standard large-rollout protocol: CALVIN uses 1000 rollouts and

each LIBERO task uses 500 trials. We do not report multi-seed training variance due to computational constraints, which we include as a limitation.

B.2 Evaluation on MIKASA-Robo

To further evaluate memory-dependent robotic manipulation, we evaluate ProRet-VLA on five tasks from the MIKASA-Robo benchmark (?). These tasks require the policy to retain and use information that is distributed across temporally separated observations. ShellGameTouch requires recalling which cup initially covered the target object, while InterceptMedium requires integrating the observed motion of a moving ball. RememberColor-3, RememberColor-5, and RememberColor-9 require recalling a previously observed target color, with the number of candidate colors increasing across the three settings.

Table 7 reports the task success rates and the action chunk size K used by each configuration. ProRet-VLA achieves the highest average success rate of 45.2%, compared with 39.6% for MemoryVLA. It obtains the best results on ShellGameTouch and RememberColor-9, achieving success rates of 89% and 22%, respectively. The gains are not uniform across all tasks: π_0 performs best on InterceptMedium, OpenVLA with $K = 8$ performs best on RememberColor-3, and MemoryVLA performs best on RememberColor-5. These results indicate that ProRet-VLA achieves strong aggregate performance across the evaluated memory-dependent settings rather than uniformly dominating every task.

The comparison is not strictly controlled with respect to action chunking. In particular, ProRet-VLA uses $K = 3$, whereas MemoryVLA uses $K = 4$, and the remaining baselines also use different values of K . Because action chunk size may affect temporal aggregation and closed-loop control, the 5.6-point average difference between ProRet-

Table 7: Performance comparison on five memory-dependent tasks from MIKASA-Robo. Success rates (%) are reported. K denotes the action chunk size used by each configuration. The best result in each column is shown in bold.

Method	K	ShellGame Touch	Intercept Medium	Remember Color-3	Remember Color-5	Remember Color-9	Avg. Success
OpenVLA	4	12	6	21	9	8	11.2
SpatialVLA	4	23	27	27	17	11	21.0
OpenVLA	8	47	14	59	16	6	28.4
π_0	4	33	42	35	22	15	29.4
MemoryVLA	4	84	17	47	30	20	39.6
ProRet-VLA	3	89	39	52	24	22	45.2

Table 8: Inference latency. We report per-step inference latency under the same evaluation setting as the main experiments. P95 denotes the 95th-percentile latency, meaning that 95% of inference steps are no slower than this value. We test 5 times and take average time. Δ denotes the relative increase of mean latency over the baseline.

Method	Mean (ms)	Δ (%)	P95 (ms)	Peak Mem. (GB)
Baseline	82.1	–	141.5	4.01
w/ Pro	88.3	+7.6	171.6	4.23
w/ Ret	84.7	+3.1	166.2	4.02
Ours	94.1	+14.6	185.1	4.23

VLA and MemoryVLA cannot be attributed entirely to the proposed architecture. We therefore interpret these results as complementary benchmark-level evidence of memory-dependent task performance.

B.3 Details of the Ablation

The *w/ Ret* variant is designed to isolate the effect of retrospective memory retrieval under the same policy backbone, action decoder, training setting, and evaluation protocol as the other ablation variants. It enables the episodic memory retrieval branch before action prediction: at each decision step, the current token $\mathbf{x}_i$ retrieves a historical counterpart $\hat{\mathbf{R}}_i$ from the memory buffer. For fusion, *w/ Ret* also adopts a learned gate rather than a fixed fusion ratio. However, the gate only takes $\mathbf{x}_i$ and $\hat{\mathbf{R}}_i$ as inputs, and predicts a token-wise weight to adaptively combine current context with retrieved memory. Unlike the full ProRet-VLA gate, it does not use the predicted deviation $\hat{d}_t$ as an input. Therefore, *w/ Ret* evaluates retrospective retrieval with feature-only adaptive fusion, without deviation-driven regulation.

Difference from MemoryVLA. The *w/ Ret* variant is not intended as a reproduction of the full MemoryVLA framework. Instead, it is a controlled internal ablation under our unified experimental setting. It only enables retrospective retrieval

Table 9: Deviation Predictor reliability. Metrics are computed on 100 evaluation batches with 6,400 samples. AUC@20/10 treats samples in the top 20%/10% of ground-truth d_t as high-deviation positives and evaluates whether $\hat{d}_t$ ranks them ahead of others.

MSE↓	Pearson↑	Spearman↑	AUC@20↑	AUC@10↑
0.000887	0.4861	0.4683	0.7464	0.7757

and feature-only gated fusion, while disabling the prospection branch, the future prediction loss, the deviation predictor, and the deviation-conditioned gate. Therefore, *w/ Ret* measures the marginal contribution of retrospective memory within our ablation system, whereas MemoryVLA represents an independent memory-centric VLA baseline with its own full model design.

B.4 Memory Hyperparameter Ablations.

Since memory-related hyperparameters may behave differently across architectures and datasets, we conduct controlled ablations on CALVIN ABC-D to validate our memory design choices. All variants use the same backbone, prospection branch, deviation-driven gate, and training protocol; only the memory configuration is changed.

As shown in Tab. 6, the dual-stream buffer outperforms the unified buffer, improving the average length from 4.41 to 4.50. This suggests that separating perceptual and cognitive memory streams helps reduce interference between high-frequency visual dynamics and relatively stable task-level semantics. For buffer length, $L = 8$ underperforms due to insufficient historical capacity, while increasing the length to $L = 32$ does not bring further gains and slightly reduces the average length to 4.46. This indicates that overly long memory may introduce redundant or stale information, whereas $L = 16$ provides a better trade-off between historical coverage and retrieval reliability. Finally, ToMe-based consolidation achieves better performance than FIFO replacement, improving the av-

Table 10: Real-world task details. We list the language instruction template, number of variations, and variation type for each real-world task.

Task Name	Language Instruction Template	Variations	Variation Type
Press Buttons in Order	“Press the buttons in order: pink, blue, yellow.”	3	Button spatial layout
Pick All Objects	“Pick up all objects and place them in the basket.”	2	Object set on the table

erage length from 4.37 to 4.50. This demonstrates that similarity-based consolidation better preserves informative historical states than simply discarding old entries, leading to more stable long-horizon execution.

B.5 Inference Latency

We further profile the inference latency of different variants under the same evaluation setting as the main experiments. All measurements are conducted on a single NVIDIA L20 GPU from an 8-GPU server. As shown in Table 8, the full ProRet-VLA model introduces only a moderate latency overhead over the baseline, while maintaining a practical inference speed for closed-loop evaluation. The memory-only variant incurs little additional peak memory, since its extra computation is mainly performed on compact latent tokens with a lightweight fusion gate. The future-image decoder is used only for training the prospection loss and is skipped during inference; therefore, the additional inference cost mainly comes from compact latent retrieval, deviation prediction, and gated fusion.

B.6 Deviation Predictor Reliability

Since the high-deviation event analysis uses the Deviation Predictor as a frozen monitor, we further evaluate whether its predicted deviation $\hat{d}_t$ aligns with the self-supervised deviation target d_t computed from future prediction error. We conduct a quick reliability check on 100 evaluation batches, containing 6,400 samples in total. As shown in Table 9, the predictor achieves a low MSE of 0.000887 and moderate positive Pearson and Spearman correlations with the target deviation. In addition, we treat high-deviation detection as a binary ranking problem, where positives are defined by the top 20% or top 10% of the ground-truth deviation values. The resulting AUC scores of 0.7464 and 0.7757 indicate that $\hat{d}_t$ is informative for identifying high-deviation states, although it is not a perfect estimator.

The predictor is not expected to exactly regress the pixel-level reconstruction error, since the target also contains visual changes that may be irrelevant

to control. Instead, our use case only requires the predicted deviation to provide a temporally aligned ranking signal for identifying phases where memory fusion should be adjusted. The AUC results indicate that the predictor is informative for high-deviation detection, which is consistent with the intervention results in Table 3. These results support its use as a frozen monitor for event-level analysis, while avoiding any effect on the action generation of baseline methods.

B.7 Case Study of Simulation.

Qualitative Results on CALVIN. Fig. 6 presents a representative ProRet-VLA rollout spanning five consecutive CALVIN instructions. The sequence involves transitions between distinct manipulation stages, illustrating how the policy maintains task progress over an extended horizon. This qualitative example is consistent with the aggregate results in Tab. 1, where ProRet-VLA achieves success rates of 98.2% and 79.9% at the one- and five-instruction horizons, respectively. DreamVLA, the closest competing method at the longer horizons, matches ProRet-VLA at one instruction with 98.2% and reaches 78.1% at five instructions. The rollout provides an illustrative example of long-horizon execution, while the controlled ablations in Sec. 4.3 provide the mechanism-level evidence for the contributions of prospection, retrospection, and deviation-conditioned memory fusion.

Qualitative Results on LIBERO. The qualitative performance on LIBERO highlights ProRet-VLA’s superior long-horizon reasoning. While strong baselines like π_0 show a significant performance drop when scaling from short-horizon tasks (98.8% on LIBERO-Object) to complex sequences (85.2% on LIBERO-Long), ProRet-VLA maintains remarkable consistency across domains. It achieves a 96.7% success rate on LIBERO-Long, outperforming π_0 by 8.7%, effectively bridging the gap between atomic actions and multi-stage planning.

This robustness is driven by the deviation-driven gating mechanism, which enables adaptability in

unstructured environments. We observe that during visual occlusions (e.g., arm blocking the target) or physical execution failures (e.g., grasp slippage), the model detects a spike in prediction deviation. This triggers an active retrieval of historical memory, allowing the agent to maintain object localization or initiate autonomous error recovery instead of continuing a failed trajectory.

B.8 Real-World Experiments

Real-World Experimental Setup. We evaluate our method on a series of long-horizon real world tasks using the Synria Alicia-D robot, which is equipped with a 6-DoF manipulator and a 1-DoF gripper. A single Intel RealSense D405 camera provides third-person observations, while an additional wrist-mounted camera of the same size provides egocentric input. Data are collected at 30 Hz. Our model is trained under the standard ProRet-VLA configuration and deployed on a single NVIDIA L20 GPU. For fair comparison, all baseline methods follow their official configurations and are also deployed on the same single NVIDIA L20 GPU.

Task Descriptions. We evaluate two real-world long-horizon manipulation tasks, as summarized in Table 10. The first task requires the robot to press three colored buttons following a fixed language-specified order: pink, blue, and yellow. To test robustness to spatial changes, we vary the physical layout of the buttons on the table while keeping the instruction unchanged. The second task requires the robot to pick up all objects on the table and place them into the basket. For this task, variations are created by changing the set of objects placed on the table. These tasks require the policy to track completed subgoals over time, such as which buttons have already been pressed or which objects have already been placed into the basket.

Visualization. Figure 7 shows representative real-world rollouts of the two long-horizon tasks. For the button-pressing task, the robot sequentially approaches and presses the pink, blue, and yellow buttons according to the fixed language instruction. For the object-collection task, the robot repeatedly grasps the remaining objects and places them into the basket until the table is cleared. The upper and lower rows in Figure 7 (b) further show two task variations with different object sets. These rollouts illustrate that successful execution requires tracking temporal task progress, such as which buttons have already been pressed or which objects have

already been placed into the basket, rather than reacting only to the current observation.

Turn off led



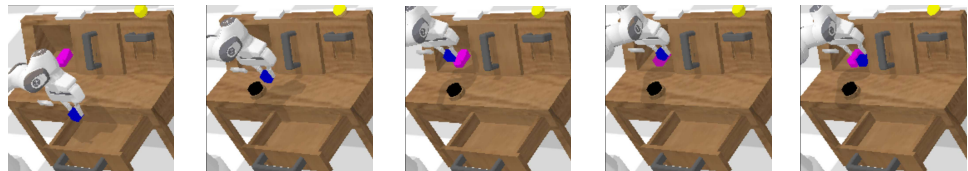
Push into drawer



Lift blue block drawer



Place in slider

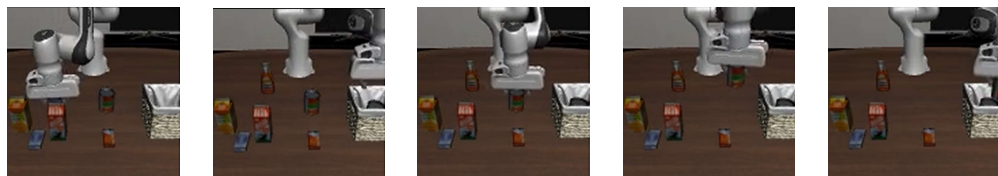


Close drawer

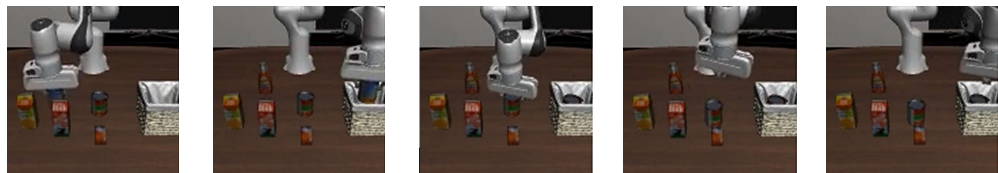


Figure 5: Qualitative results of the CALVIN long horizon tasks

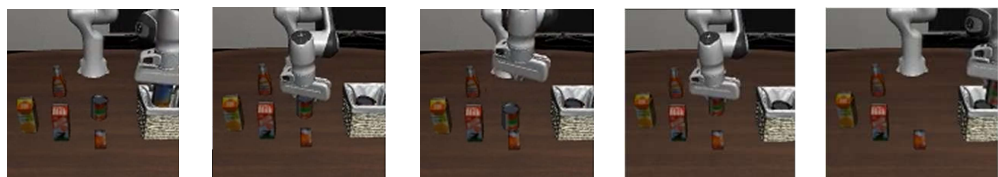
SUCCESS



FAIL

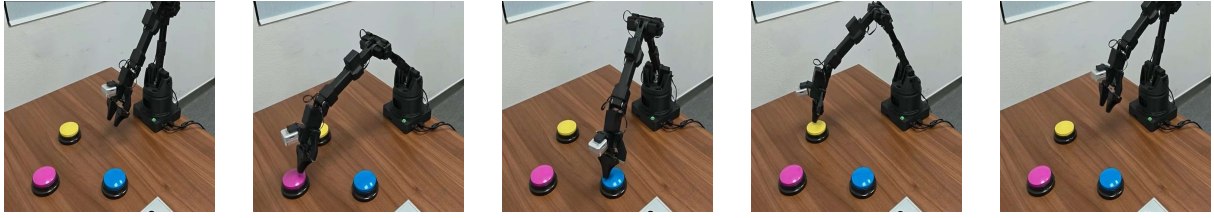


SUCCESS
(RECOVERY)

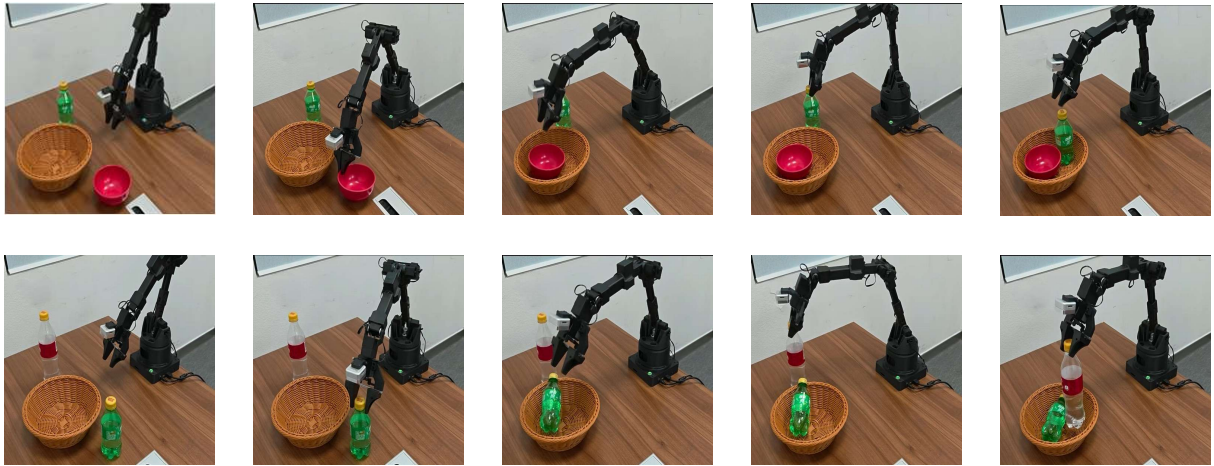


Task: put both the alphabet soup and the tomato sauce in the basket

Figure 6: Qualitative results of the Libero-long tasks



(a) Press the buttons in order: pink, blue, yellow.



(b) Pick up all objects and place them in the basket.

Figure 7: Real-world rollout visualizations. We show representative execution trajectories for the two real-world long-horizon tasks. (a) The robot presses the buttons according to the specified order: pink, blue, and yellow. (b) The robot sequentially picks up all objects and places them into the basket, where the upper and lower rows correspond to two task variations with different object sets. Frames are ordered from left to right, illustrating the need to track completed subgoals over time.