SCAR: SUPERVISED, COMPRESSED, AND ALIGNED REPRESENTATIONS FOR MEMORY-AUGMENTED VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

In long-horizon robotic manipulation, vision-language-action (VLA) policies must retain task-relevant information that may not be observable from the current input. Recent works on perceptual memory approach this problem by conditioning VLAs on past observations, but learning effective memory representations remains challenging. Action prediction provides only sparse supervision for information that must be retained over long horizons, raw observation histories contain temporal redundancy, and newly introduced memory representations are often underutilized by the pretrained policies. To address these challenges, we introduce **SCAR** (**S**upervised, **C**ompressed, and **A**ligned memory **R**epresentations), which combines textual supervision of memory-relevant events, learned temporal compression, and staged memory alignment. We evaluate SCAR on 16 memory-dependent manipulation tasks from the RoboMME benchmark, spanning temporal, spatial, object, and procedural memory. SCAR improves average success rate from 38.0% to 48.0% over the closest architectural perceptual-memory baseline, with controlled ablations demonstrating the contribution of all three components. Across four real-world manipulation tasks, SCAR further improves average success from 30.8% to 51.7% over the same baseline. These results demonstrate that explicitly supervising, compressing, and aligning perceptual memory improves memory-dependent manipulation in both simulation and real-world settings.

1 INTRODUCTION

Long-horizon robotic manipulation remains a challenge for imitation-learning-based policies such as vision-language-action (VLA) models (Black et al., 2025a; Bjorck et al., 2025). A key limitation of current VLAs is their implicit assumption of full observability: the entire task-relevant state is inferred from the current proprioceptive and visual observations together with the task prompt. This assumption breaks in partially observable environments, where correct actions depend on information that is no longer directly observable (Kaelbling et al., 1998; Sutton & Barto, 2018). For example, when wiping a table, a robot must remember which areas have already been cleaned.

A natural solution is to augment VLAs with memory. Recent approaches summarize the interaction history through external VLMs (Sridhar et al., 2026; Torne et al., 2026), recurrently maintain latent memory states (Jiang et al., 2026; Cherepanov et al., 2026; Li et al., 2026a), or directly encode sequences of past observations as perceptual memory (Jang et al., 2025; Koo et al., 2026). We focus on *perceptual memory*, which avoids the communication bottleneck of VLM-based summaries and the optimization challenges of recurrent approaches (Dai et al., 2026). However, its effective integration introduces three challenges.

First, existing perceptual-memory approaches largely supervise memory through downstream action prediction (Jang et al., 2025; Koo et al., 2026). Because many actions can be inferred from the current observation alone, this provides only sparse supervision for information that must persist over long horizons. Memory representations should therefore receive explicit supervision for semantically relevant historical information. Second, conditioning on long visual histories introduces substantial redundancy, particularly across neighboring frames (Villasevil et al., 2025; Mark et al., 2026). Yet, these frames also contain dynamic information about motion and state transitions. Ef-

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.

fective memory must therefore aggregate temporally local observations while suppressing redundant visual content. Third, memory introduces a new modality into a pretrained VLA whose parameters are already specialized to the original inputs. Fine-tuning can therefore converge to solutions that underutilize the new memory pathway (Lee et al., 2026), motivating an explicit alignment strategy.

To address these challenges, we propose SCAR, which learns **s**upervised, **c**ompressed, and **a**ligned memory **r**epresentations through three methodological contributions. First, we introduce an auxiliary objective that predicts textual summaries of memory-relevant events, providing a denser learning signal for memory. Second, a temporal compression module reduces the memory representation along the temporal axis while imposing an inductive bias toward compact representations of temporally local information. Third, we employ a two-stage training curriculum that first aligns the memory pathway with the pretrained VLA before jointly fine-tuning the complete model.

We instantiate SCAR on a Mixture-of-Transformers VLA and evaluate it on the RoboMME benchmark (Dai et al., 2026), comprising 16 manipulation tasks across four memory categories. SCAR outperforms existing memory-augmented VLA approaches and improves on the closest architectural perceptual-memory baseline by 10 percentage points. Systematic ablations isolate the contributions of auxiliary supervision, temporal compression, and staged alignment. Finally, across four real-world manipulation tasks, SCAR improves on the closest perceptual-memory baseline by 20.9 percentage points, demonstrating that the improvements observed in simulation transfer to physical robot settings.

In summary, we introduce SCAR, a memory-augmented VLA framework combining explicit semantic supervision, temporal compression, and staged memory alignment, and we evaluate its effectiveness across 16 simulated memory tasks and four real-world robotic manipulation tasks.

2 Related Work

Mixture-of-Transformer VLAs. Recent VLA models adapt pretrained VLMs for robotic control through large-scale post-training. Action generation is integrated through approaches such as autoregressive decoding (Zitkovich et al., 2023; Kim et al., 2025) or separate diffusion models conditioned on VLM representations (Bjorck et al., 2025). We focus on *Mixture-of-Transformers* (MoT) architectures, popularized by the π family (Black et al., 2026; 2025a; Amin et al., 2026; Intelligence et al., 2026) and adopted by subsequent VLAs (Fang et al., 2026; Wu et al., 2026; Wang et al., 2026). In MoT-VLAs, visual and language tokens are processed by a high-capacity VLM backbone, while continuous action tokens are handled by a smaller action expert trained with diffusion or flow matching; both streams interact through layer-wise joint self-attention.

Memory in Vision-Language-Action Models. Recent memory-augmented VLA approaches can be broadly grouped by memory representation. (1) *Explicit state representations* use an external model, typically a VLM, to reason over observations, selected keyframes, or textual task state, and produce subgoal instructions (Sridhar et al., 2026; Torne et al., 2026; Team et al., 2025) or keyframes (Mark et al., 2026), which condition the VLA. These approaches perform well in event-salient tasks such as counting, but lack the spatio-temporal fidelity required for object identity or motion cues (Dai et al., 2026). (2) *Recurrent memory* aggregates history into a fixed-size latent state, either through test-time training of fast weights (Jiang et al., 2026) or recurrently updated memory tokens (Cherepanov et al., 2026; Li et al., 2026a). (3) *Perceptual memory* directly conditions the policy on visual history, as in RoboMME *FrameSamp+Expert* (Dai et al., 2026), and ContextVLA (Jang et al., 2025); HAMLET (Koo et al., 2026) additionally enriches visual representations with task context into stored memory tokens. Perceptual memory preserves richer spatio-temporal information, but must identify sparse relevant events within highly redundant histories under indirect action supervision.

Auxiliary Supervision for VLA and Memory Learning. Language-based auxiliary supervision has long been used in VLAs. RT-2 (Zitkovich et al., 2023) co-trains action prediction with Visual Question Answering, while $\pi_{0.5}$ (Black et al., 2025a) additionally predicts subtasks and shows that subtask prediction can benefit the policy even when omitted at inference. More recent embodied Chain-of-Thought approaches (Zawalski et al., 2025; Sun et al., 2025; 2026) supervise high-level plans, low-level motions, and grounded information such as object or end-effector locations.

Because autoregressive reasoning increases inference latency, ECOT-Lite (Chen et al., 2025) and ZR0 (Li et al., 2026b) use reasoning traces only during training, suggesting that language supervision can improve internal representations without requiring inference-time reasoning. Auxiliary objectives for memory-augmented VLAs have been less explored. PTP (Villasevil et al., 2025) predicts past actions to reduce spurious temporal correlations, while ReMem-VLA (Li et al., 2026a) reconstructs a past observation. However, these objectives do not explicitly identify which historical information is task-relevant. In contrast, our summary-and-subtask objective directly uses semantically relevant past events for supervision.

Integrating and Compressing Memory Representations. Long observation histories require efficient temporal compression. RoboMME *TokenDrop* (Dai et al., 2026) removes temporally redundant visual tokens based on RGB differences, but this can reduce spatial expressiveness on tasks requiring precise spatial relationships. In contrast, SCAR uses a temporal convolution to aggregate neighboring observations into compact temporal representations.

Introducing a new modality into a pretrained model often benefits from staged optimization. LLaVA (Liu et al., 2023) first aligns visual features with a pretrained language model before joint fine-tuning, while MoSS (Lee et al., 2026) applies a similar strategy to tactile and force-torque signals in VLAs. We adopt this principle for memory, but find that fully freezing the pretrained policy during alignment can degrade manipulation performance, motivating our soft alignment.

3 Preliminaries

We consider a MoT-VLA policy $\pi(A_t \mid O_t, \ell)$, where the corresponding current observation $O_t = (\boldsymbol{I}_t^1, \ldots, \boldsymbol{I}_t^{N_V}, q_t)$ consists of RGB images $\boldsymbol{I}_t^v \in \mathbb{R}^{H\times W\times 3}$ from N_V camera views and the current proprioceptive state q_t. Conditioned on O_t and task instruction ℓ, the policy generates an action chunk $A_t = (a_t, \ldots, a_{t+K-1}) \sim \pi(\cdot \mid O_t, \ell)$ of K commands via flow matching (Lipman et al., 2023). Because the policy only observes the current state, it has no explicit representation of preceding interactions or temporal dynamics such as its previous direction of motion.

3.1 Baseline Architecture

To provide historical context, we augment a pretrained MoT-VLA with episodic perceptual memory. Specifically, we build on RoboMME *FrameSamp+Expert* (Dai et al., 2026), resulting in the memory-augmented policy $\tilde{\pi}(A_t \mid O_t, \ell, \mathcal{H}_t)$, where $\mathcal{H}_t = \{\boldsymbol{I}_0, \ldots, \boldsymbol{I}_t\}$ denotes the visual history from a single camera view. Given $\mathcal{H}_t$, *FrameSamp+Expert* selects up to N_F frames at evenly spaced temporal locations, $F_t \in \mathbb{R}^{N_F\times H\times W\times 3}$. Consequently, the temporal spacing between samples increases as the episode progresses. Each frame is independently encoded by the vision encoder, spatially pooled, augmented with 3D spatio-temporal positional embeddings, and linearly projected to obtain $Z_t \in \mathbb{R}^{N_Z\times d}$, where N_Z is the flattened memory-token sequence length. Memory is integrated through an additional memory expert initialized from scratch, while the remaining components inherit their pretrained parameters. In the transformer layer i, the backbone and action-expert tokens attend to the corresponding memory representation Z_t^i produced by the memory expert.

3.2 Problem Definition

Our objective is to learn memory representations Z_t^i that (1) preserve decision-relevant information from $\mathcal{H}_t$ while (2) discarding redundant perceptual content and aggregating local temporal dynamics. We express this through the conceptual information-bottleneck objective

$$\max_{Z_t^i} \underbrace{I(Z_t^i; E_t \mid \ell)}_{\substack{\text{preserve}\\ \text{relevant events}}} - \beta \underbrace{I(Z_t^i; \mathcal{H}_t \mid \ell)}_{\substack{\text{compress}\\ \text{perceptual history}}}, \tag{1}$$

where $I(\cdot;\cdot)$ denotes the mutual information between two random variables, and E_t denotes an abstract semantic representation of task-relevant historical events, including completed task steps, previously observed object identities or locations, and demonstrated motion patterns. Conceptually, conditioning an optimal policy on E_t should provide the same decision-relevant historical information as conditioning on the complete history $\mathcal{H}_t$. We use Eq. 1 only as a conceptual objective and do not estimate either mutual-information term directly.

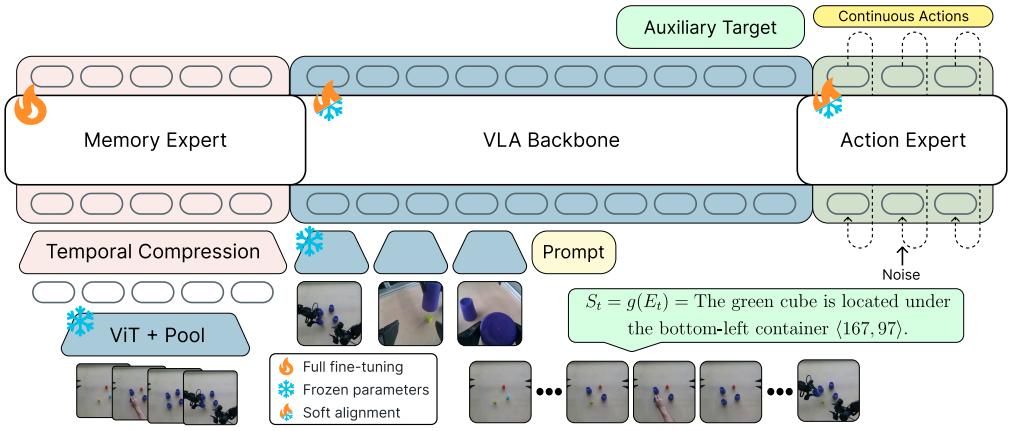

Figure 1: SCAR augments a memory-based MoT-VLA with (1) textual supervision of memory-relevant events, (2) learned temporal compression of perceptual history, and (3) staged alignment of the new memory pathway with the pretrained policy.

Sparse Supervision of Memory-Relevant Information. Memory-augmented VLAs are typically trained through an action prediction objective. Consequently, the action target A_t is the primary learning signal for shaping the internal memory representations Z_t^i, while supervising the encoding of relevant memory events E_t only indirectly. This supervision is weak whenever the current observation already contains sufficient information to predict the next action. For many time steps,

$$\tilde{\pi}(A_t \mid O_t, \ell, \mathcal{H}_t) \approx \pi(A_t \mid O_t, \ell). \tag{2}$$

In such states, memory contributes little additional action-relevant information, $I(Z_t^i; A_t \mid O_t, \ell) \approx 0$. As a result, action prediction alone only sparsely constrains which historical information should be preserved in Z_t^i, motivating an additional objective that directly supervises learning of memory-relevant events.

Redundancy in Perceptual History. Preserving short-duration events requires high temporal resolution, but increasing N_F also introduces many highly correlated neighboring frames with little additional task-relevant information. In terms of equation 1, increasing temporal resolution makes satisfying the compression term more challenging: the representation should preserve relevant temporal information without unnecessarily increasing its dependence on redundant details of $\mathcal{H}_t$. This motivates explicitly compressing neighboring observations into compact temporal representations.

4 Methodology

4.1 Textual Summaries as Direct Memory Supervision

To address sparse action supervision, we augment the flow-matching objective with an auxiliary textual objective that directly supervises the memory representation. We define a target $S_t = g(E_t)$ that summarizes task-relevant memory events and follows the structured format "*Demonstration: ...; Summary: ...; Subtask: ...*". *Demonstration* describes the relevant context from a provided demonstration, *Summary* records relevant events during live execution, and *Subtask* specifies the next high-level subgoal. Targets are generated from the simulator state or human annotations. The model is co-trained to predict S_t using the autoregressive cross-entropy objective

$$\mathcal{L}_{\text{aux}} = -\mathbb{E}\left[\log p_\theta(S_t \mid Z_t, \ell)\right]. \tag{3}$$

Minimizing Eq. 3 reduces the entropy $H(S_t \mid Z_t, \ell)$ and therefore encourages the memory representation to retain the predictive information of the textual target:

$$I(Z_t; S_t \mid \ell) = H(S_t \mid \ell) - H(S_t \mid Z_t, \ell). \tag{4}$$

Since $S_t = g(E_t)$ is constructed directly to encode task-relevant information from the memory events E_t, maximizing information about S_t provides a practical surrogate for encouraging the relevance term $I(Z_t^i; E_t \mid \ell)$ in equation 1. Importantly, we use this information-theoretic interpretation

only as motivation; mutual-information terms are not explicitly estimated during training. The resulting training objective combines flow-matching with the auxiliary summary objective as

$$\mathcal{L} = \mathcal{L}_{\mathrm{FM}} + \lambda_{\mathrm{aux}} \mathcal{L}_{\mathrm{aux}}, \tag{5}$$

where $\lambda_{\mathrm{aux}} \geq 0$ controls the auxiliary-loss weight. Auxiliary language tokens are appended to the VLA-backbone sequence and can attend only to the task prompt ℓ, memory tokens Z_t, and preceding auxiliary tokens. They cannot attend to the current observation O_t, preventing the model from solving the auxiliary task through information available only in the current visual observation. Conversely, memory, backbone, and action tokens cannot attend to the auxiliary tokens. Thus, textual supervision is used only during training and introduces no autoregressive language-generation overhead at inference. The complete attention mask is provided in Appendix A.2.

4.2 Temporal Compression

Figure 1 illustrates our extension of a pretrained MoT-VLA with memory. Following section 3.1 we introduce memory tokens into the VLA using a dedicated memory expert. To provide this memory pathway with a temporally structured input, we reconsider how observations are sampled from the episode history. We hypothesize that a regular temporal structure facilitates learning local temporal dependencies and prevents the temporal semantics of neighboring memory frames from changing over the course of an episode. Therefore, we sample at most N_F observations from $\mathcal{H}_t = \{\boldsymbol{I}_0, \ldots, \boldsymbol{I}_t\}$ using a fixed temporal stride ΔT_F:

$$F_t = \mathrm{StrideSample}\left(\mathcal{H}_t; \Delta T_F, N_F\right). \tag{6}$$

This bounds the maximum observation horizon to $T_{\max} = (N_F - 1)\Delta T_F$, but provides consistent temporal spacing between neighboring observations irrespective of the current episode length. Histories shorter than N_F are right-padded, yielding $F'_t \in \mathbb{R}^{N_F \times H \times W \times 3}$. Each valid frame is encoded by the pretrained vision encoder, and its patch embeddings are spatially pooled to obtain the memory tokens $M_t \in \mathbb{R}^{N_F \times H' \times W' \times d}$.

We then apply a temporal compression module consisting of 1D convolutions independently at each spatial token location. With total temporal stride s, the module reduces the temporal dimension to $N_M = N_F/s$ and produces compressed memory tokens $M'_t \in \mathbb{R}^{N_M \times H' \times W' \times d}$. Its temporal receptive field aggregates neighboring observations into compact spatio-temporal representations, providing an inductive bias toward preserving informative temporal changes while suppressing redundant content. In terms of Eq. 1, this bottleneck is designed to reduce unnecessary dependence on $\mathcal{H}_t$ while retaining memory-relevant information.

Finally, M'_t is flattened, augmented with 3D spatio-temporal positional embeddings, and projected into $Z_t \in \mathbb{R}^{N_Z \times d}$, where $N_Z = N_M H' W'$. These memory tokens are integrated into the VLA using the memory expert. Further details of the implementation are provided in Appendix A.1.

4.3 Soft Memory Alignment

The memory expert introduces randomly initialized parameters into an otherwise pretrained VLA. Initially, its representations Z_t^i are therefore misaligned with pretrained backbone or action expert. Finetuning can already minimize much of the action loss using the informative context (O_t, ℓ), providing little incentive to incorporate the initially uninformative memory pathway. As a result, the model may converge to a solution in which the utilization of Z_t^i remains small. We therefore introduce a two-stage curriculum that allows the memory pathway to adapt before jointly fine-tuning the complete policy. During the first T_{align} steps, the memory expert uses the full learning rate η_{mem}, while the pretrained backbone and the action expert are updated with

$$\eta_{\mathrm{backbone}}(t) = \eta_{\mathrm{action}}(t) = \alpha_t \eta_{\mathrm{mem}}, \qquad \alpha_t : 0 \to \alpha_{\mathrm{align}} < 1.$$

This allows rapid adaptation of the memory pathway while permitting limited updates to the pretrained policy. In the second stage, α_t is increased from α_{align} to 1 and all components are jointly fine-tuned at the same learning rate. To reduce optimization instability at the transition between stages, we linearly increase the learning-rate scaling factor from α_{align} to 1 at the beginning of the joint fine-tuning stage. Completely freezing the pretrained components during alignment can degrade manipulation precision; α_{align} therefore controls a trade-off between preserving pretrained capabilities and adapting the policy to the new memory representation.

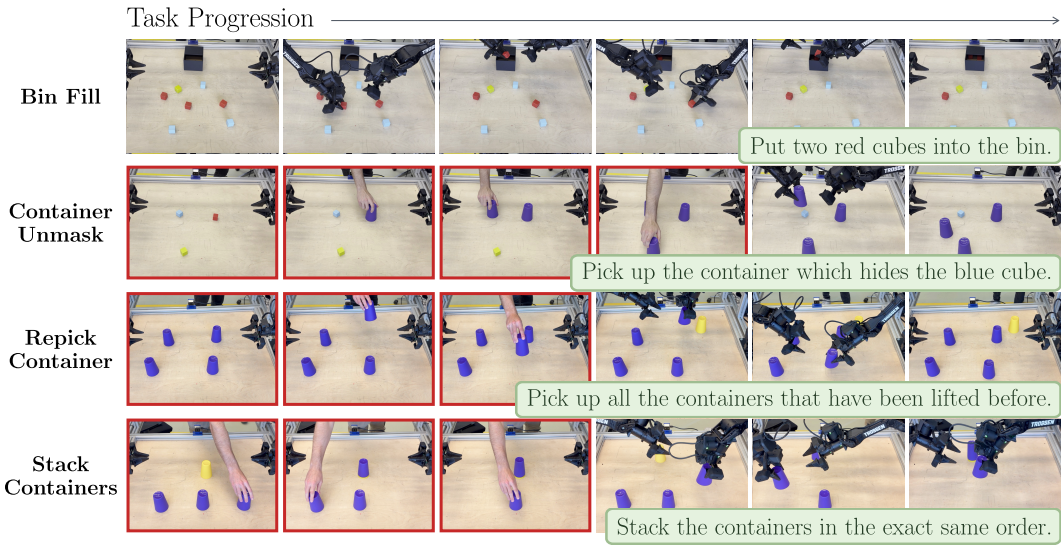

Figure 2: Overview of real-world tasks. Red borders: demo frames; green boxes: task prompts.

5 Experiments

We evaluate our proposed methodology on memory-dependent robotic manipulation tasks in simulation and the real world. Our experiments are designed to answer the following research questions:

- RQ1: How does SCAR perform compared to existing memory-augmented VLA approaches?
- RQ2: What supervision information best supports perceptual memory learning?
- RQ3: Does temporal compression improve memory beyond more history or model capacity?
- RQ4: Does staged memory alignment improve memory integration into pretrained VLAs?
- RQ5: Do the improvements observed in simulation persist in real-world manipulation?

5.1 Evaluation Setup

Implementation Details. We instantiate SCAR using the $\pi_{0.5}$ MoT-VLA (Black et al., 2025a) architecture. Both current observations and historical memory frames are encoded using the pretrained SigLIP vision encoder. For computational efficiency, we keep the vision encoder frozen and precompute the visual embeddings before policy training. The temporal compression module uses a temporal stride of 4, resulting in a receptive field of 13 input frames. Relative to the $\sim$ 3B-parameter pretrained policy, the compression module and memory expert add 50M and 190M parameters, respectively (240M total, $\sim 8\%$). Further architectural details are provided in Appendix A.3.

Simulation Setup. We evaluate on the RoboMME benchmark (Dai et al., 2026), comprising 16 tasks that span temporal, spatial, object, and procedural memory. These tasks require tracking past events, retaining spatial or object information, or reproducing previously demonstrated behaviors. For some tasks, memory is initialized from a video demonstration. RoboMME builds on ManiSkill (Tao et al., 2025) and uses a 7-DoF Franka Panda with 256×256 third-person and wrist RGB observations. Following RoboMME, we omit the proprioceptive state and construct episodic memory exclusively from the third-person camera. We generate 100 training episodes per task and derive *Demonstration*, *Summary*, and *Subtask* targets from the simulator state; more details are provided in Appendix A.4. We evaluate the final training checkpoint across 50 episodes and four model seeds.

Real-World Setup. We design four RoboMME-inspired tasks, shown in Figure 2, each targeting one memory type: *Bin Fill* evaluates temporal memory by requiring the robot to track previously completed placements, *Container Unmask* tests spatial memory through object localization under occlusion, *Repick Container* evaluates object memory by requiring consistent object identification

Table 1: RQ1 – Comparison with existing memory-augmented VLA approaches on RoboMME. Success rates (%) are reported as mean ± standard deviation across four seeds.

Method	Temporal				Spatial			
	Bin Fill	Pick Xtimes	Swing Xtimes	Stop Cube	Video Unmask	Button Unmask	Video Unmask Swap	Button Unmask Swap
FS+Expert	$\mathbf{50.0}^{\pm 3.7}$	$92.0^{\pm 3.2}$	$\mathbf{96.0}^{\pm 1.4}$	$42.5^{\pm 5.0}$	$23.5^{\pm 1.7}$	$26.0^{\pm 2.4}$	$20.5^{\pm 5.0}$	$19.0^{\pm 2.2}$
FS+Modul	$43.5^{\pm 1.7}$	$\mathbf{95.0}^{\pm 3.0}$	$84.0^{\pm 5.1}$	$40.5^{\pm 6.2}$	$23.5^{\pm 2.6}$	$32.5^{\pm 1.7}$	$\mathbf{21.5}^{\pm 4.6}$	$\mathbf{22.5}^{\pm 3.8}$
SCAR	$40.5^{\pm 3.8}$	$\mathbf{95.0}^{\pm 1.7}$	$75.5^{\pm 0.9}$	$\mathbf{58.5}^{\pm 3.6}$	$\mathbf{51.5}^{\pm 3.0}$	$\mathbf{50.5}^{\pm 1.7}$	$15.0^{\pm 3.0}$	$22.0^{\pm 2.4}$
$\pi_{0.5}$	$27.0^{\pm 2.2}$	$39.5^{\pm 4.8}$	$35.5^{\pm 3.8}$	$2.5^{\pm 1.7}$	$24.5^{\pm 0.9}$	$22.0^{\pm 1.4}$	$17.5^{\pm 1.7}$	$11.0^{\pm 4.4}$
MemER	$66.0^{\pm 3.2}$	$89.0^{\pm 3.6}$	$74.5^{\pm 3.8}$	$3.0^{\pm 3.0}$	$94.5^{\pm 2.6}$	$64.5^{\pm 18.2}$	$30.0^{\pm 2.0}$	$20.5^{\pm 6.7}$

Method	Object				Procedural				AVG
	Pick Highlight	Video Repick	Video Place Button	Video Place Order	Move Cube	Insert Peg	Pattern Lock	Route Stick	
FS+Expert	$19.0^{\pm 3.0}$	$25.5^{\pm 2.2}$	$32.5^{\pm 2.6}$	$23.0^{\pm 3.6}$	$\mathbf{85.5}^{\pm 3.6}$	$1.5^{\pm 1.7}$	$14.5^{\pm 1.7}$	$37.0^{\pm 1.7}$	$38.0^{\pm 0.6}$
FS+Modul	$19.0^{\pm 4.6}$	$30.0^{\pm 1.4}$	$54.5^{\pm 3.8}$	$32.5^{\pm 4.3}$	$76.0^{\pm 4.5}$	$6.5^{\pm 1.7}$	$\mathbf{51.0}^{\pm 5.7}$	$\mathbf{66.0}^{\pm 3.2}$	$43.7^{\pm 0.6}$
SCAR	$\mathbf{38.0}^{\pm 1.4}$	$\mathbf{43.0}^{\pm 1.7}$	$\mathbf{68.5}^{\pm 1.7}$	$\mathbf{36.0}^{\pm 1.4}$	$82.0^{\pm 2.0}$	$\mathbf{12.0}^{\pm 4.0}$	$22.5^{\pm 3.8}$	$58.0^{\pm 3.2}$	$\mathbf{48.0}^{\pm 0.8}$
$\pi_{0.5}$	$12.5^{\pm 1.7}$	$1.5^{\pm 1.7}$	$19.5^{\pm 0.9}$	$17.5^{\pm 3.6}$	$26.0^{\pm 2.4}$	$2.5^{\pm 1.7}$	$9.5^{\pm 3.0}$	$6.5^{\pm 2.2}$	$17.2^{\pm 0.6}$
MemER	$77.5^{\pm 3.8}$	$33.5^{\pm 2.2}$	$31.0^{\pm 6.1}$	$26.5^{\pm 3.8}$	$77.0^{\pm 3.6}$	$3.5^{\pm 0.9}$	$24.0^{\pm 4.7}$	$12.5^{\pm 3.3}$	$45.5^{\pm 0.8}$

across demonstration and execution, and *Stack Containers* tests procedural memory by requiring the robot to reproduce a demonstrated manipulation sequence. For each task, we collect 100 training episodes using a bimanual Trossen WidowX AI leader–follower setup. The auxiliary targets are human annotated, as detailed in Appendix A.4. Episodic memory is constructed exclusively from the third-person camera stream. Further details are provided in Appendix A.5.

5.2 Evaluation Results

RQ1: How does SCAR perform compared to existing memory-augmented VLA approaches? We compare SCAR against four baselines: (1) RoboMME *FrameSamp+Expert*, the architecture most closely related to our method (Sec. 3.1); (2) RoboMME *FrameSamp+Modul*, which integrates perceptual memory through adaptive Layer Normalization (AdaLN) within the action expert; (3) MemER (Sridhar et al., 2026), a non-perceptual memory baseline that uses a VLM to select keyframes and to produce a subgoal which conditions the VLA; and (4) the $\pi_{0.5}$ policy without episodic memory. All models are trained for 80k optimization steps. SCAR uses a cosine-decaying schedule, as we find that Soft Memory Alignment benefits from a high learning rate during the initial alignment phase followed by gradual adaptation during joint fine-tuning. In contrast, the original RoboMME training recipe uses a constant learning rate. We analyze the effect of these scheduling choices in Appendix B.1.

Table 1 shows that SCAR achieves the highest average success rate, exceeding the perceptual memory baselines *FrameSamp+Expert* by 10.0 pp and *FrameSamp + Modul* by 4.3 pp. All memory-based methods substantially outperform $\pi_{0.5}$, highlighting the importance of historical context in partially observable environments. MemER is particularly strong on *Video Unmask* and *Pick Highlight*, demonstrating the benefits of explicitly retaining keyframes and leveraging VLM-based reasoning. In contrast, perceptual memory methods perform better on *Stop Cube*, *Video Place Button*, *Pattern Lock*, and *Route Stick*, which all require reasoning over dynamic scene changes. SCAR gains most on *Object* memory and several *Spatial* tasks, while all methods struggle on the swap variants, possibly due to spatial compression discarding fine-grained information required to track the container-swapping dynamics. *FrameSamp+Modul* is strongest on several motion-centric procedural tasks, due to its more direct modulation of the action-generation pathway, which facilitates

the transfer of temporally structured motion information into action prediction. Finally, SCAR underperforms on the *Temporal* tasks *BinFill* and *SwingXtimes*. We suspect that this may reflect the trade-off between memory usage and dexterity introduced by soft memory alignment (Sec. 4.3).

RQ2: What supervision information best supports perceptual memory learning? Starting from the full SCAR architecture, we vary the auxiliary objective: *Action only* removes auxiliary supervision; *Subtask* predicts the next high-level subgoal; *Summary* predicts memory-relevant past events; *Subtask+Summary* combines both; and *Subtask+Summary** additionally removes pixel-location annotations. All variants are trained on the 16 RoboMME tasks for 40k optimization steps.

Figure 3 shows that, overall, auxiliary supervision outperforms action-only training. The full *Subtask+Summary* target increases average success from 36.9% (*Action only*) to 46.4%. *Subtask* reaches 39.9%, whereas *Summary* achieves 43.3%, suggesting that explicitly supervising *what should be remembered* is more informative than predicting only the next subgoal. Combining both yields the highest overall performance, while removing spatial grounding reduces success to 44.8%, indicating that past events, future task progress, and spatial grounding provide complementary supervision. All memory types follow this trend except *Temporal*, where the targets perform similarly, suggesting that action supervision is already sufficient for these tasks. Overall, these results support our hypothesis that perceptual memory benefits from selective supervision of past events. Full per-task results are provided in Appendix B.2.

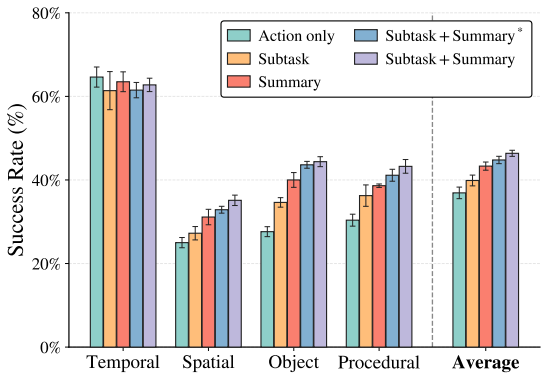

Figure 3: RQ2 – Comparison of auxiliary memory supervision targets averaged across tasks.

RQ3: Does temporal compression improve memory beyond more history or model capacity? Next, we disentangle the effect of temporal compression from two potential confounding factors: access to a longer observation history and the additional model capacity introduced by the compression module. We compare four variants: *32F*, using 32 frames without compression; *128F*, extending history to 128 uncompressed frames; *32F+Capacity*, adding a parameter-matched non-temporal module; and *128F+TC*, the full SCAR temporal-compression model. All remaining architectural and training settings are fixed.

Figure 4 illustrates the effect of each design factor. Increasing history from 32 to 128 frames improves average success from 39.2% to 41.0% (+1.8 pp), while parameter matching reaches 41.8% (+2.6 pp). Temporal compression instead raises the 128-frame model to 46.4% (+5.4 pp). This

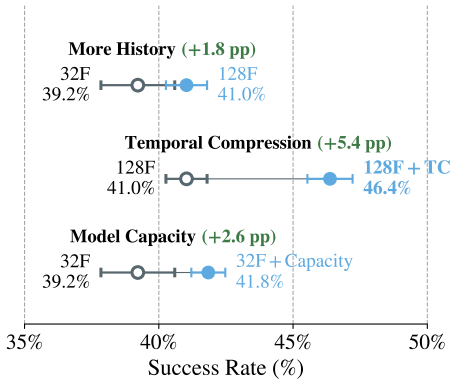

Figure 4: RQ3 – Isolating temporal compression, history length and model capacity.

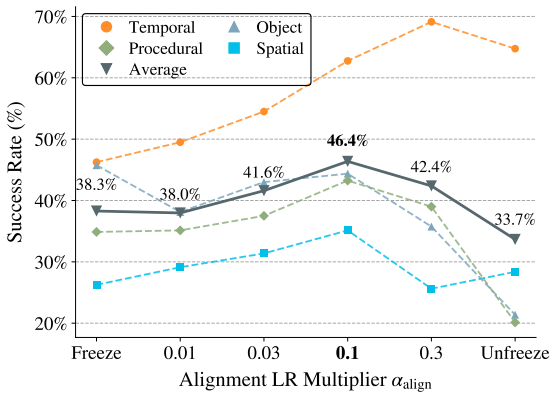

Figure 5: RQ4 – Effect of alignment strength on memory integration and policy performance.

Table 2: RQ5 – Real-world evaluation results. Mean success rates (%) across episodes.

Model	Bin Fill	Container Unmask	Repick Container	Stack Containers	AVG
$\pi_{0.5}$	26.7%	33.3%	13.3%	20.0%	23.3%
FS+Expert	50.0%	30.0%	13.3%	30.0%	30.8%
SCAR	**70.0%**	**36.7%**	**36.7%**	**63.3%**	**51.7%**

performance gain therefore cannot be explained by history length or parameter count alone and supports the hypothesis of explicitly aggregating neighboring observations into compact temporal representations. Full per-task results are provided in Appendix B.3.

RQ4: Does staged memory alignment improve memory integration into pretrained VLAs? To evaluate the influence of different alignment choices on policy performance, we sweep α_{align}, which scales the learning rates of the pretrained backbone and action expert relative to the newly introduced memory expert during the initial alignment stage. We additionally evaluate *Freeze* ($\alpha_{\text{align}} = 0$) and *Unfreeze*, which removes the alignment stage. All variants are trained on the 16 RoboMME tasks for 40k optimization steps.

Figure 5 shows that our intermediate setting, $\alpha_{\text{align}} = 0.1$, achieves the highest performance at 46.4%, compared with 38.3% for *Freeze* and 33.7% for *Unfreeze*. These results support the motivation for soft memory alignment: the memory representations should first adapt to the pretrained VLA before jointly optimizing all parameters. At the same time, overly restrictive freezing can degrade the manipulation capabilities of the pretrained policy. The optimal intermediate scaling therefore reflects a trade-off between integrating the new memory representation and preserving manipulation capabilities acquired during pretraining. Interestingly, the temporal memory category peaks at $\alpha_{\text{align}} = 0.3$ and appears to benefit less from memory alignment than the other categories; rollout inspection suggests that soft alignment hurts manipulation precision on *BinFill* and *SwingXtimes*. Full per-task results are provided in Appendix B.4 and we additionally analyze the attention mass under different alignment schedules in Appendix B.5.

RQ5: Do the improvements observed in simulation persist in real-world manipulation? We compare SCAR with memory-free $\pi_{0.5}$ and *FrameSamp+Expert* on the four real-world tasks introduced in Sec. 5.1. Each method trains for 30k steps on 100 demonstrations per task and is evaluated over 30 rollouts per task.

Table 2 reports the real-world success rates. SCAR achieves 51.7% average success, compared with 30.8% for *FrameSamp+Expert* and 23.3% for $\pi_{0.5}$. The largest gains occur on *Bin Fill*, *Repick Container*, and *Stack Containers*. Task-level trends do not transfer identically from simulation: for example, SCAR underperforms *FrameSamp+Expert* on simulated *Bin Fill* but clearly outperforms it in the real world. Conversely, SCAR's clear advantage on simulated spatial-memory tasks is much less pronounced on *Container Unmask*. A possible explanation is that RoboMME provides several structurally related spatial tasks, yielding more shared training experience than the single real-world counterpart. Overall, SCAR's advantage persists in physical manipulation despite task-specific differences between simulation and the real world.

6 Conclusion

We present SCAR, a framework for learning supervised, compressed, and aligned perceptual memory representations in vision-language-action models. On RoboMME, SCAR improves over the closest architectural perceptual-memory baseline by 10.0 percentage points. Our ablations show that explicit supervision of memory-relevant events substantially improves performance, temporal compression provides gains beyond increased history length or model capacity, and intermediate staged alignment outperforms both full freezing and immediate joint fine-tuning. These benefits persist in real-world manipulation, where SCAR reaches 51.7% average success compared with 30.8% for the closest perceptual-memory baseline. Overall, our results show that effective perceptual memory requires not only access to longer histories, but representations explicitly shaped to retain task-relevant information, aggregate temporal context, and integrate effectively with pretrained VLA policies.

AI USE STATEMENT

In this work, we used generative AI tools to support the development of theoretical models or conceptual frameworks, the formulation of mathematical claims, the refinement of hypotheses, the design of and feedback on research methodology and experiments, the implementation of methods, and the cleaning and reformatting of datasets. We did not use generative AI tools for proposing hypotheses, assisting with translation, interpreting results, or generating synthetic datasets. The use of generative AI tools for providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, and supporting qualitative or thematic data analysis was not applicable to this work. Additionally, we used generative AI tools for creating or modifying scientific figures or images, suggesting experimental parameters, creating or editing software code, brainstorming, editing the research paper to improve readability, identifying relevant literature, and proposing a title or keywords for the research paper. We reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness. Conceptual ideas generated by LLMs were evaluated for novelty and checked against the existing literature through manual literature review. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide detailed descriptions of the model architecture, training hyperparameters, and experimental setup in the appendix, together with additional experimental results and precise definitions of the textual auxiliary supervision targets. The source code required to reproduce our method and experiments is available at https://anonymous.4open.science/r/scar-3975.

REFERENCES

Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace B. Connors, James Darpinian, Karan Dhabalia, Jared Di Carlo, Danny Driess, Michael Robert Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Benjamin Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin Leblanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alexander Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi_{0.6}$: a VLA That Learns From Experience. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026. doi: 10.15607/RSS.2026.XXII.087.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi, 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. *CoRR*, abs/2503.14734, March 2025. URL https://doi.org/10.48550/arXiv.2503.14734.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *9th Annual Conference on Robot Learning*, 2025a. URL https://openreview.net/forum?id=vlhoswksBO.

Kevin Black, Manuel Y Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=UkR2zO5uww.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2026. URL https://arxiv.org/abs/2410.24164.

William Chen, Suneel Belkhale, Suvir Mirchandani, Karl Pertsch, Danny Driess, Oier Mees, and Sergey Levine. Training strategies for efficient embodied reasoning. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 365–391. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/chen25a.html.

Egor Cherepanov, Nikita Kachaev, Daniil Zelezetsky, Aydar Bulatov, Artem Pshenitsyn, Yuri Kuratov, Alexey Skrynnik, Aleksandr I. Panov, and Alexey K. Kovalev. μVLA: On recurrent memory for partially observable manipulation in vla models, 2026. URL https://arxiv.org/abs/2606.12497.

Yinpei Dai, Hongze Fu, Jayjun Lee, Yuejiang Liu, Haoran Zhang, Jianing Yang, Chelsea Finn, Nima Fazeli, and Joyce Chai. RoboMME: Benchmarking and understanding memory for robotic generalist policies. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=8m30ogkPk2.

Haoquan Fang, Jiafei Duan, Donovan Clay, Sam Wang, Shuo Liu, Weikai Huang, Xiang Fan, Wei-Chuan Tsai, Shirui Chen, Yi Ru Wang, Shanli Xing, Jaemin Cho, Jae Sung Park, Ainaz Eftekhar, Peter Sushko, Karen Farley, Angad Wadhwa, Cole Harrison, Winson Han, Ying-Chun Lee, Eli VanderBilt, Rose Hendrix, Suveen Ellawela, Lucas Ngoo, Joyce Chai, Zhongzheng Ren, Ali Farhadi, Dieter Fox, and Ranjay Krishna. MolmoAct2: Action reasoning models for real-world deployment, 2026. URL https://arxiv.org/abs/2605.02881.

Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, Vedant Choudhary, Foster Collins, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Maitrayee Dhaka, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachlan Groom, Haroun Habeeb, Hunter Hancock, Karol Hausman, Gashon Hussein, Victor Hwang, Brian Ichter, Connor Jacobsen, Szymon Jakubczak, Rowan Jen, Tim Jones, Gregg Kammerer, Ben Katz, Liyiming Ke, Mairbek Khadikov, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Brendon LeCount, Sergey Levine, Xinyu Li, Adrian Li-Bell, Vladislav Lialin, Zhonglin Liang, Wallace Lim, Yao Lu, Enyu Luo, Vishnu Mano, Nandan Marwaha, Aikys Mongush, Liam Murphy, Suraj Nair, Tyler Patterson, Karl Pertsch, Allen Z. Ren, Gavin Schelske, Charvi Sharma, Baifeng Shi, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Jiaming Tang, Jimmy Tanner, Shalom Tekeste, Marcel Torne, Kyle Vedder, Quan Vuong, Anna Walling, Haohuan Wang, Jason Wang, XuDong Wang, Chris Whalen, Samuel Whitmore, Blake Williams, Charles Xu, Sukwon Yoo, Lili Yu, Wuming Zhang, Zhuoyang Zhang, and Ury Zhilinsky. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities, 2026. URL https://arxiv.org/abs/2604.15483.

Huiwon Jang, Sihyun Yu, Heeseung Kwon, Hojin Jeon, Younggyo Seo, and Jinwoo Shin. ContextVLA: Vision-language-action model with amortized multi-frame context, 2025. URL https://arxiv.org/abs/2510.04246.

Yunfan Jiang, Yevgen Chebotar, Ruijie Zheng, Fengyuan Hu, Yunhao Ge, Jimmy Wu, Tianyuan Dai, Scott Reed, Li Fei-Fei, Yuke Zhu, and Linxi "Jim" Fan. RoboTTT: Context scaling for robot policies, 2026. URL https://arxiv.org/abs/2607.15275.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134, 1998. ISSN 0004-3702. doi: 10.1016/S0004-3702(98)00023-X.

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

Myungkyu Koo, Daewon Choi, Taeyoung Kim, Kyungmin Lee, Changyeon Kim, Younggyo Seo, and Jinwoo Shin. HAMLET: Switch your vision-language-action model into a history-aware policy. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=KcJ9U0x6kO.

Jimin Lee, Huiwon Jang, Myungkyu Koo, Jungwoo Park, and Jinwoo Shin. Modular sensory stream for integrating physical feedback in vision-language-action models, 2026. URL https://arxiv.org/abs/2604.23272.

Hang Li, Fengyi Shen, Dong Chen, Liudi Yang, Xudong Wang, Jinkui Shi, Zhenshan Bing, Ziyuan Liu, and Alois Knoll. ReMem-VLA: Empowering vision-language-action model with memory via dual-level recurrent queries, 2026a. URL https://arxiv.org/abs/2603.12942.

Haoyang Li, Guanlin Li, Youhe Feng, Chen Zhao, Zhuoran Wang, Yang Li, Qizhe Wei, Shifeng Bao, Haitao Shen, Yihan Zhao, Tong Yang, and Jing Zhang. Training vision-language-action models with dense embodied chain-of-thought supervision, 2026b. URL https://arxiv.org/abs/2606.30552.

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

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=w0H2xGHlkw.

Max Sobol Mark, Maria Attarian, Chuyuan Fu, Debidatta Dwibedi, Jacky Liang, Dhruv Shah, and Aviral Kumar. Long-Context Robot Imitation Learning by Focusing on Key History Frames. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026. doi: 10.15607/RSS.2026.XXII.201.

Ajay Sridhar, Jennifer Pan, Satvik Sharma, and Chelsea Finn. Scaling up memory for robotic control via experience retrieval. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=1dH4ARGdwD.

Nan Sun, Yuan Zhang, Yongkun Yang, Wentao Zhao, Peiyan Li, Jun Guo, Wenxuan Song, Pengxiang Ding, Runze Suo, Yifei Su, Xin Xiao, Xinghang Li, and Huaping Liu. Revisiting embodied chain-of-thought for generalizable robot manipulation, 2026. URL https://arxiv.org/abs/2606.03784.

Qi Sun, Pengfei Hong, Tej Deep Pala, Vernon Toh, U-Xuan Tan, Deepanway Ghosal, and Soujanya Poria. Emma-X: An embodied multimodal action model with grounded chain of thought and look-ahead spatial reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14199–14214, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.695. URL https://aclanthology.org/2025.acl-long.695/.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh N, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. ManiSkill3: GPU parallelized robot simulation and rendering for generalizable embodied AI. In *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025. URL https://openreview.net/forum?id=GgTxudXaU8.

Gemini Robotics Team, Abbas Abdolmaleki, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Ashwin Balakrishna, Nathan Batchelor, Alex Bewley, Jeff Bingham, Michael Bloesch, Konstantinos Bousmalis, Philemon Brakel, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Christine Chan, Oscar Chang, London Chappellet-Volpini, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, Adrian Collister, David B. D'Ambrosio, Sudeep Dasari, Todor Davchev, Meet Kirankumar Dave, Coline Devin, Norman Di Palo, Tianli Ding, Carl Doersch, Adil Dostmohamed, Yilun Du, Debidatta Dwibedi, Sathish Thoppay Egambaram, Michael Elabd, Tom Erez, Xiaolin Fang, Claudio Fantacci, Cody Fong, Erik Frey, Chuyuan Fu, Ruiqi Gao, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Oliver Groth, Agrim Gupta, Roland Hafner, Steven Hansen, Leonard Hasenclever, Sam Haves, Nicolas Heess, Brandon Hernaez, Alex Hofer, Jasmine Hsu, Lu Huang, Sandy H. Huang, Atil Iscen, Mithun George Jacob, Deepali Jain, Sally Jesmonth, Abhishek Jindal, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Matija Kecman, J. Chase Kew, Donnie Kim, Frank Kim, Junkyung Kim, Thomas Kipf, Sean Kirmani, Ksenia Konyushkova, Li Yang Ku, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Tuan Anh Le, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Guy Lever, Jacky Liang, Li-Heng Lin, Fangchen Liu, Shangbang Long, Caden Lu, Sharath Maddineni, Anirudha Majumdar, Kevis-Kokitsi Maninis, Andrew Marmon, Sergio Martinez, Assaf Hurwitz Michaely, Niko Milonopoulos, Joss Moore, Robert Moreno, Michael Neunert, Francesco Nori, Joy Ortiz, Kenneth Oslund, Carolina Parada, Emilio Parisotto, Amaris Paryag, Acorn Pooley, Thomas Power, Alessio Quaglino, Haroon Qureshi, Rajkumar Vasudeva Raju, Helen Ran, Dushyant Rao, Kanishka Rao, Isaac Reid, David Rendleman, Krista Reymann, Miguel Rivas, Francesco Romano, Yulia Rubanova, Peter Pastor Sampedro, Pannag R Sanketi, Dhruv Shah, Mohit Sharma, Kathryn Shea, Mohit Shridhar, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricut, Rachel Sterneck, Ian Storz, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Saran Tunyasuvunakool, Jake Varley, Grace Vesom, Giulia Vezzani, Maria Bauza Villalonga, Oriol Vinyals, René Wagner, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Chengda Wu, Markus Wulfmeier, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Jimmy Yan, Sherry Yang, Skye Yang, Yuxiang Yang, Hiu Hong Yu, Wenhao Yu, Wentao Yuan, Yuan Yuan, Jingwei Zhang, Tingnan Zhang, Zhiyuan Zhang, Allan Zhou, Guangyao Zhou, and Yuxiang Zhou. Gemini Robotics 1.5: Pushing the frontier of generalist robots with advanced embodied reasoning, thinking, and motion transfer, 2025. URL https://arxiv.org/abs/2510.03342.

Marcel Torne, Karl Pertsch, Homer Walke, Kyle Vedder, Suraj Nair, Brian Ichter, Allen Z. Ren, Haohuan Wang, Jiaming Tang, Kyle Stachowicz, Karan Dhabalia, Michael Equi, Quan Vuong, Jost Tobias Springenberg, Sergey Levine, Chelsea Finn, and Danny Driess. MEM: Multi-scale embodied memory for vision language action models, 2026. URL https://arxiv.org/abs/2603.03596.

Marcel Torne Villasevil, Andy Tang, Yuejiang Liu, and Chelsea Finn. Learning long-context diffusion policies via past-token prediction. In *9th Annual Conference on Robot Learning*, 2025. URL https://openreview.net/forum?id=o0LBjJxUeS.

Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, Siteng Huang, Yifan Tang, Wenhui Wang, Ru Zhang, Jianyi Liu, and Donglin Wang. Vla-adapter: an effective paradigm for tiny-scale vision-language-action model. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence and Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence and Sixteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'26/IAAI'26/EAAI'26. AAAI Press, 2026. ISBN 978-1-57735-906-7. doi: 10.1609/aaai.v40i22.38931. URL https://doi.org/10.1609/aaai.v40i22.38931.

Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, Yiyu Ren, Kejia Zhang, Hui Yu, Jingmei Zhao, Shuai Zhou, Zhenqi Qiu, Houlong Xiong, Ziyu Wang, Zechen Wang, Ran Cheng, Yong-Lu Li, Yongtao Huang, Xing Zhu, Yujun Shen, and Kecheng Zheng. A pragmatic vla foundation model, 2026. URL https://arxiv.org/abs/2601.18692.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of

Proceedings of Machine Learning Research, pp. 3157–3181. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/zawalski25a.html.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

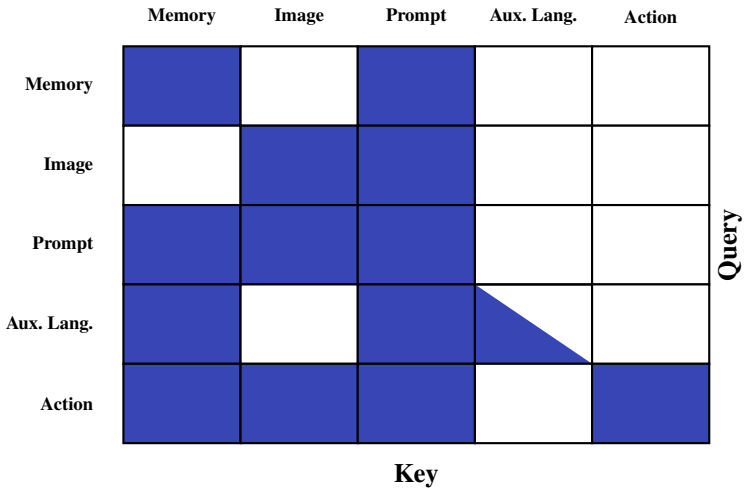

Figure 6: Segment-level attention masking pattern. Colored regions denote permitted attention between query and key token groups. Auxiliary-language tokens use causal self-attention, indicated by the lower-triangular region.

A Implementation Details

A.1 Temporal Compression Module

Given memory tokens $M_t \in \mathbb{R}^{N_F \times H' \times W' \times d}$, we reduce the temporal resolution while preserving the spatial resolution and feature dimension using a stack of $L = \log_2(N_F/N_M)$ residual temporal convolution blocks. Each block applies a stride-2 temporal convolution independently at every spatial location:

$$M_t^{(l+1)} = \mathrm{Conv}_{1,2}^{(l)}\left(M_t^{(l)}\right) + W^{(l)}\,\mathrm{GELU}\left(\mathrm{Conv}_{k,2}^{(l)}\left(\mathrm{LN}\left(M_t^{(l)}\right)\right)\right),$$

where $W^{(l)}$ is a linear layer, $\mathrm{Conv}_{k,2}$ denotes a 1D temporal convolution with kernel size k and stride 2, and $\mathrm{Conv}_{1,2}$ forms the strided residual projection. Starting from $M_t^{(0)} = M_t$, the temporal dimension is halved in each block, yielding the compressed representation $M_t' = M_t^{(L)} \in \mathbb{R}^{N_M \times H' \times W' \times d}$.

A.2 Attention Masking Pattern

Figure 6 illustrates the block attention mask used in our SCAR architecture. The auxiliary language tokens follow a block-causal attention pattern and can attend only to the prompt and memory tokens, but not to the current image observation. Conversely, no other token type is allowed to attend to the auxiliary language tokens. This isolates the auxiliary language pathway from the policy computation, allowing SCAR to leverage language-based supervision during training while removing the associated computational overhead at inference time.

A.3 Implementation Details and Training Hyperparameters

All models (SCAR and RoboMME) use a maximum memory-token budget of 512 tokens for the memory expert. This corresponds to a maximum of $N_F = 32$ frames for RoboMME and $N_F = 128$ frames for SCAR. RoboMME samples frames using an evenly spaced sampling scheme, whereas SCAR employs the strided sampling strategy defined in equation 6. For episodes exceeding the maximum observation horizon $T_{\max} = (N_F - 1)\Delta T_F$, SCAR discards observations older than this horizon. We choose the temporal sampling stride ΔT_F such that even the longest episodes remain fully contained within the resulting observation window.

Table 3: Training and optimization hyperparameters used for the RoboMME & $\pi_{0.5}$ baselines and SCAR. Values that differ between simulation (Sim.) and real-world experiments (Real) are reported separately.

Hyperparameter	**RoboMME & $\pi_{0.5}$**	**SCAR**
Environment and policy		
FPS	20 (Sim.), 50 (Real)	20 (Sim.), 50 (Real)
Action horizon	20 (Sim.), 50 (Real)	20 (Sim.), 50 (Real)
Proprioception	No (Sim.), Yes (Real)	No (Sim.), Yes (Real)
Optimization		
Batch size	64	64
Optimizer	AdamW	AdamW
β_1	0.9	0.9
β_2	0.95	0.95
Weight decay	10^{-10}	10^{-10}
EMA decay	0.999	0.999
LR schedule	Constant	Cosine decay
Learning rate	5×10^{-5}	1×10^{-4}
Final learning rate	–	1×10^{-5}
Warmup steps	–	1,000
Training duration		
RQ1	80k	80k
RQ2–RQ4	–	40k
Real-world	30k	30k

Table 4: SCAR-specific memory and alignment hyperparameters.

Hyperparameter	**Value**
Maximum number of memory frames N_F	128
Temporal sampling stride (frames) ΔT_F	10 (Sim.), 20 (Real)
Auxiliary loss weight λ_{aux}	0.05
Alignment duration T_{align}	20k (RQ1–RQ4), 15k (Real)
Alignment LR multiplier α_{align}	0.1

Both methods encode each image independently using the $\pi_{0.5}$ SigLIP encoder, producing features of shape $N_F \times 16 \times 16 \times 2048$. Spatial pooling then reduces these features to $N_F \times 4 \times 4 \times 2048$. For SCAR, the resulting memory representation is subsequently compressed along the temporal dimension using the 1D convolution module described in Appendix A.1. We instantiate this module with two temporal convolution blocks and a kernel size of $k = 5$, resulting in a temporal reduction factor of 4.

The resulting SCAR and RoboMME representations are then flattened to 512×2048, combined with 3D spatio-temporal positional embeddings, and projected to the final memory-token sequence of shape 512×1024.

Common training hyperparameters for SCAR, RoboMME, and $\pi_{0.5}$ are reported in Table 3, while SCAR-specific hyperparameters are provided in Table 4.

A.4 Auxiliary Supervision Fields

We summarize the information encoded in the *Summary* and *Demonstration* fields of the auxiliary language targets S_t for the RoboMME (Dai et al., 2026) tasks in Table 5 and for the real-world tasks in Table 6. If either field is not described for a task, it is omitted from that task's target. The *Subtask* field simply specifies the next subgoal to be executed by the robot and is therefore not discussed further.

Table 5: Auxiliary supervision fields used for the RoboMME simulation tasks.

Task	Auxiliary supervision
BinFill	*Summary* records completed placement events, e.g., "Placed the red cube into the bin."
PickXtimes	*Summary* records completed pick-and-place events together with their repetition count, e.g., "Picked and placed the red cube one time."
SwingXtimes	*Summary* records how many times the robot has visited the right or left target, e.g., "Moved the cube over the right target one time."
StopCube	*Summary* records how many times the cube has passed over the target, e.g., "The cube has passed the target one time."
VideoUnmask	*Demonstration* maps each container to its hidden cube using relative identities and locations, e.g., "The bottom-left container $\langle 127, 89\rangle$ hides the blue cube; the top container $\langle 86, 143\rangle$ hides the red cube; and the rightmost container $\langle 123, 177\rangle$ hides the green cube."
ButtonUnmask	*Summary* records which colored cubes are hidden by which containers together with their locations, e.g., "The red cube was covered by the top-right container $\langle 82, 157\rangle$, the green cube by the bottom container $\langle 125, 160\rangle$, and the blue cube by the leftmost container $\langle 95, 99\rangle$."
VideoUnmaskSwap	*Demonstration* records the post-swap container-to-cube associations and locations, e.g., "The blue cube is covered by the top-left container $\langle 88, 78\rangle$, the green cube by the middle container $\langle 97, 140\rangle$, and the red cube by the bottom-right container $\langle 102, 180\rangle$." *Summary* records the executed container-lifting event.
ButtonUnmaskSwap	*Summary* records the current cube-to-container associations and their locations, e.g., "The blue cube is covered by the leftmost container $\langle 94, 92\rangle$, the green cube by the bottom container $\langle 96, 130\rangle$, and the red cube by the top-right container $\langle 90, 165\rangle$."
PickHighlight	*Summary* identifies the previously highlighted target object by color and relative position, e.g., "The highlighted cube was the middle red cube."
VideoRepick	*Demonstration* identifies the manipulated cube by color, relative position, and location, e.g., "The manipulated cube is the rightmost blue cube $\langle 102, 179\rangle$."
VideoPlaceButton	*Demonstration* records the demonstrated placement target and its relation to the button press, e.g., "The cube was placed on the rightmost target $\langle 134, 175\rangle$ after the button was pressed."
VideoPlaceOrder	*Demonstration* records the demonstrated target location and placement order, e.g., "The cube was placed on the top-right target $\langle 80, 165\rangle$ for the second time."
MoveCube	*Demonstration* describes the demonstrated relation between pushing the cube and reaching the target, e.g., "The cube was pushed with the peg onto the target."
InsertPeg	*Demonstration* describes the demonstrated peg insertion, e.g., "Picked the leftmost peg at the far end and inserted it into the right hole."
PatternLock	*Summary* records the executed sequence of directional movement, e.g., "Moved left."
RouteStick	*Summary* records the executed sequence of movements, e.g., "Moved left clockwise."

A.5 Real-World Experimental Setup

This section provides additional details on the real-world tasks, observation and action spaces, and rollout setup used for physical robot evaluation.

`Bin Fill` (*temporal memory*) presents between three and nine red, blue, or green cubes on a table. The task prompt instructs the robot to place between one and three cubes into a bin. Because previously placed cubes are no longer visible once inside the bin, the robot must track completed placements to determine when the task is finished and which colors have already been placed.

`Container Unmask` (*spatial memory*) presents three differently colored cubes that are covered by three or four visually identical containers during a human demonstration, with the fourth container serving as a decoy. The robot subsequently has to uncover the container hiding a specified colored cube, requiring it to retain the target object's location despite occlusion.

`Repick Container` (*object memory*) contains three or four visually identical containers. During a human demonstration, between one and three containers are lifted or displaced. The robot must

Table 6: Auxiliary supervision fields used for the real-world tasks.

Task	Auxiliary supervision
Bin Fill	*Summary* records completed cube-placement events, e.g., "Placed the red cube into the bin."
Container Unmask	*Demonstration* records the demonstrated mapping between cube identities and their hiding containers together with image-space locations, e.g., "The red cube is located under the middle container $\langle 113, 111 \rangle$, the blue cube under the bottom-left container $\langle 133, 74 \rangle$, and the green cube under the top-right container $\langle 91, 150 \rangle$."
Repick Container	*Demonstration* records the identities and locations of the containers manipulated during the demonstration, e.g., "The bottom-left container $\langle 143, 78 \rangle$ was lifted." *Summary* tracks the number of lifted containers during execution, e.g., "Robot lifted the first container."
Stack Containers	*Demonstration* records the demonstrated stacking order, e.g., "First the left container, then the middle container, and finally the right container were placed on the stack."

subsequently manipulate the same containers, although not necessarily in the same order, requiring it to preserve object identities from demonstration through execution.

`Stack Containers` (*procedural memory*) consists of three visually identical purple containers arranged in a row and a fourth yellow container positioned in front of them. The robot observes a human demonstration in which the purple containers are stacked onto the yellow container in a specific order and must subsequently reproduce the demonstrated stacking sequence.

The observation space consists of three 256×256 RGB views—one third-person camera and two wrist-mounted cameras—together with a 14-dimensional proprioceptive state comprising 12 arm-joint values and two gripper states. As in simulation, episodic memory is constructed exclusively from the third-person camera stream. The 14-dimensional action space consists of the corresponding joint and gripper commands.

During inference, the memory-augmented policy exhibits an average inference latency of approximately 200ms, compared with approximately 70ms for the $\pi_{0.5}$ baseline. To maintain smooth closed-loop control despite this increased latency, we employ Real-Time Chunking (RTC) (Black et al., 2025b) during all real-robot rollouts. RTC mitigates temporal inconsistencies between observation acquisition and action execution, thereby reducing the rollout distribution shift induced by inference delays.

B Additional Experiments

B.1 RQ1 – RoboMME Learning Rate Scheduling

Table 7 compares the RoboMME *FrameSamp+Expert* baseline under our cosine-decay learning-rate schedule (*FS+Expert**) and the original constant learning-rate schedule used by RoboMME (*FS+Expert*); the corresponding hyperparameters are reported in Table 3. The original RoboMME schedule performs better on 12 of the 16 tasks and improves the overall success rate from 18.5% to 38.0%. This shows that directly transferring our learning-rate schedule to the RoboMME baseline substantially degrades its performance and motivates evaluating each baseline with its intended optimization procedure.

We hypothesize that the constant RoboMME schedule provides an optimization benefit related to the motivation behind our staged alignment strategy in Sec. 4.3. In contrast, our cosine schedule begins with a comparatively large learning rate, which may cause the pretrained backbone and action expert to adapt before the newly initialized memory pathway has learned useful representations. The resulting performance is close to that of the memory-free $\pi_{0.5}$ baseline in Table 1, which is consistent with our hypothesis of insufficient utilization of the memory pathway. SCAR mitigates this issue by applying the high initial learning rate primarily to the memory expert during the alignment stage T_{align}, while reducing the learning rates of the pretrained backbone and action expert.

Table 7: Effect of the learning-rate schedule on the *FrameSamp+Expert* baseline. Results are reported as success rate (%) with standard deviation across seeds. *FS+Expert** uses our cosine-decay learning-rate schedule, whereas *FS+Expert* uses the original constant learning-rate schedule from RoboMME.

Method	Temporal				Spatial			
	Bin Fill	Pick Xtimes	Swing Xtimes	Stop Cube	Video Unmask	Button Unmask	Video Unmask Swap	Button Unmask Swap
FS+Expert*	$26.5^{\pm 2.6}$	$38.0^{\pm 3.7}$	$33.0^{\pm 5.7}$	$3.0^{\pm 3.3}$	$\mathbf{26.0}^{\pm 1.4}$	$\mathbf{26.5}^{\pm 8.3}$	$16.5^{\pm 2.2}$	$9.5^{\pm 5.2}$
FS+Expert	$\mathbf{50.0}^{\pm 3.7}$	$\mathbf{92.0}^{\pm 3.2}$	$\mathbf{96.0}^{\pm 1.4}$	$\mathbf{42.5}^{\pm 5.0}$	$23.5^{\pm 1.7}$	$26.0^{\pm 2.4}$	$\mathbf{20.5}^{\pm 5.0}$	$\mathbf{19.0}^{\pm 2.2}$

Method	Object				Procedural				AVG
	Pick Highlight	Video Repick	Video Place Button	Video Place Order	Move Cube	Insert Peg	Pattern Lock	Route Stick	
FS+Expert*	$13.0^{\pm 1.7}$	$3.5^{\pm 4.1}$	$26.0^{\pm 1.4}$	$\mathbf{24.0}^{\pm 2.4}$	$29.0^{\pm 3.0}$	$\mathbf{2.5}^{\pm 2.2}$	$10.0^{\pm 1.4}$	$9.0^{\pm 2.2}$	$18.5^{\pm 0.6}$
FS+Expert	$\mathbf{19.0}^{\pm 3.0}$	$\mathbf{25.5}^{\pm 2.2}$	$\mathbf{32.5}^{\pm 2.6}$	$23.0^{\pm 3.6}$	$\mathbf{85.5}^{\pm 3.6}$	$1.5^{\pm 1.7}$	$\mathbf{14.5}^{\pm 1.7}$	$\mathbf{37.0}^{\pm 1.7}$	$\mathbf{38.0}^{\pm 0.6}$

Table 8: RQ2 – Full per task evaluation results. *Subt.+Sum.* refers to Subtask+Summary and *Subt.+Sum.** to Subtask+Summary* without spatial location labels. We report mean success rate (%) with standard deviation across four seeds.

Method	Temporal				Spatial			
	Bin Fill	Pick Xtimes	Swing Xtimes	Stop Cube	Video Unmask	Button Unmask	Video Unmask Swap	Button Unmask Swap
Action only	$\mathbf{49.0}^{\pm 1.0}$	$\mathbf{97.5}^{\pm 1.7}$	$\mathbf{79.0}^{\pm 5.4}$	$33.0^{\pm 5.7}$	$33.5^{\pm 1.7}$	$25.5^{\pm 0.9}$	$18.5^{\pm 5.5}$	$\mathbf{22.5}^{\pm 3.0}$
Subtask	$38.5^{\pm 7.4}$	$93.5^{\pm 2.2}$	$74.5^{\pm 3.0}$	$39.0^{\pm 7.7}$	$39.0^{\pm 2.2}$	$29.5^{\pm 6.7}$	$20.0^{\pm 1.4}$	$20.5^{\pm 3.0}$
Summary	$35.5^{\pm 3.8}$	$93.0^{\pm 1.0}$	$75.0^{\pm 1.0}$	$\mathbf{50.5}^{\pm 5.2}$	$41.5^{\pm 4.8}$	$46.5^{\pm 1.7}$	$17.0^{\pm 2.2}$	$19.5^{\pm 3.8}$
Subt.+Sum.*	$40.5^{\pm 4.6}$	$96.0^{\pm 0.0}$	$64.0^{\pm 3.7}$	$45.5^{\pm 5.9}$	$\mathbf{46.5}^{\pm 1.7}$	$\mathbf{48.0}^{\pm 4.2}$	$20.5^{\pm 1.7}$	$16.5^{\pm 1.7}$
Subt.+Sum.	$44.0^{\pm 2.0}$	$87.5^{\pm 0.9}$	$72.5^{\pm 5.9}$	$47.0^{\pm 3.3}$	$45.5^{\pm 1.7}$	$45.5^{\pm 3.0}$	$\mathbf{32.5}^{\pm 3.8}$	$17.0^{\pm 1.7}$

Method	Object				Procedural				AVG
	Pick Highlight	Video Repick	Video Place Button	Video Place Order	Move Cube	Insert Peg	Pattern Lock	Route Stick	
Action only	$21.0^{\pm 5.2}$	$35.5^{\pm 5.7}$	$29.0^{\pm 1.0}$	$25.0^{\pm 3.6}$	$83.0^{\pm 5.4}$	$4.5^{\pm 0.9}$	$9.0^{\pm 1.0}$	$25.0^{\pm 2.2}$	$36.9^{\pm 1.4}$
Subtask	$32.5^{\pm 3.6}$	$39.0^{\pm 2.2}$	$43.0^{\pm 1.0}$	$24.0^{\pm 2.4}$	$\mathbf{84.5}^{\pm 4.8}$	$7.0^{\pm 3.3}$	$17.0^{\pm 2.2}$	$36.5^{\pm 3.6}$	$39.9^{\pm 1.3}$
Summary	$22.0^{\pm 1.4}$	$\mathbf{46.5}^{\pm 1.7}$	$56.5^{\pm 3.6}$	$35.0^{\pm 4.4}$	$82.0^{\pm 2.0}$	$4.0^{\pm 1.4}$	$21.5^{\pm 3.0}$	$47.0^{\pm 2.2}$	$43.3^{\pm 1.0}$
Subt.+Sum.*	$\mathbf{36.5}^{\pm 4.6}$	$36.0^{\pm 1.4}$	$64.0^{\pm 2.4}$	$\mathbf{38.0}^{\pm 2.4}$	$78.5^{\pm 1.7}$	$5.0^{\pm 2.2}$	$24.5^{\pm 0.9}$	$56.5^{\pm 2.2}$	$44.8^{\pm 0.9}$
Subt.+Sum.	$34.5^{\pm 2.6}$	$39.0^{\pm 3.6}$	$\mathbf{69.0}^{\pm 3.0}$	$35.0^{\pm 2.2}$	$80.0^{\pm 3.2}$	$\mathbf{8.0}^{\pm 2.8}$	$\mathbf{27.5}^{\pm 3.0}$	$\mathbf{57.5}^{\pm 2.6}$	$\mathbf{46.4}^{\pm 0.7}$

B.2 RQ2 – Per Task Results

We report the complete evaluation results per task for the RQ2 experiments in Table 8.

B.3 RQ3 – Per Task Results

We report the complete evaluation results per task for the RQ3 experiments in Table 9.

Table 9: RQ3 – Full per task evaluation results. We report mean success rate (%) with standard deviation across four seeds.

Method	Temporal				Spatial			
	Bin Fill	Pick Xtimes	Swing Xtimes	Stop Cube	Video Unmask	Button Unmask	Video Unmask Swap	Button Unmask Swap
32F	$\mathbf{48.5}^{\pm 3.8}$	$90.0^{\pm 4.2}$	$80.0^{\pm 3.2}$	$40.0^{\pm 5.8}$	$26.5^{\pm 3.8}$	$28.5^{\pm 0.9}$	$19.0^{\pm 1.0}$	$19.5^{\pm 6.2}$
128F	$36.5^{\pm 0.9}$	$\mathbf{96.0}^{\pm 2.4}$	$55.0^{\pm 1.0}$	$\mathbf{47.5}^{\pm 3.8}$	$31.5^{\pm 1.7}$	$39.0^{\pm 1.0}$	$18.0^{\pm 2.8}$	$\mathbf{22.0}^{\pm 4.0}$
32F+Cap.	$47.5^{\pm 2.6}$	$92.0^{\pm 3.2}$	$\mathbf{89.0}^{\pm 2.2}$	$39.0^{\pm 1.0}$	$29.0^{\pm 3.0}$	$26.5^{\pm 3.8}$	$27.0^{\pm 2.2}$	$17.0^{\pm 3.3}$
128F+TC	$44.0^{\pm 2.0}$	$87.5^{\pm 0.9}$	$72.5^{\pm 5.9}$	$47.0^{\pm 3.3}$	$\mathbf{45.5}^{\pm 1.7}$	$\mathbf{45.5}^{\pm 3.0}$	$\mathbf{32.5}^{\pm 3.8}$	$17.0^{\pm 1.7}$

Method	Object				Procedural				AVG
	Pick Highlight	Video Repick	Video Place Button	Video Place Order	Move Cube	Insert Peg	Pattern Lock	Route Stick	
32F	$21.0^{\pm 1.0}$	$36.0^{\pm 5.1}$	$62.0^{\pm 1.4}$	$24.5^{\pm 3.0}$	$74.5^{\pm 9.1}$	$\mathbf{8.0}^{\pm 2.8}$	$18.0^{\pm 1.4}$	$31.5^{\pm 2.2}$	$39.2^{\pm 1.2}$
128F	$\mathbf{41.5}^{\pm 0.9}$	$35.5^{\pm 4.8}$	$43.0^{\pm 4.4}$	$28.5^{\pm 0.9}$	$72.5^{\pm 0.9}$	$\mathbf{8.0}^{\pm 2.4}$	$\mathbf{30.0}^{\pm 1.4}$	$52.0^{\pm 2.4}$	$41.0^{\pm 0.7}$
32F+Cap.	$28.0^{\pm 5.8}$	$34.5^{\pm 3.8}$	$61.5^{\pm 2.6}$	$\mathbf{35.0}^{\pm 2.2}$	$\mathbf{81.5}^{\pm 2.2}$	$\mathbf{8.0}^{\pm 2.0}$	$17.0^{\pm 1.7}$	$37.0^{\pm 1.7}$	$41.8^{\pm 0.5}$
128F+TC	$34.5^{\pm 2.6}$	$\mathbf{39.0}^{\pm 3.6}$	$\mathbf{69.0}^{\pm 3.0}$	$\mathbf{35.0}^{\pm 2.2}$	$80.0^{\pm 3.2}$	$\mathbf{8.0}^{\pm 2.8}$	$27.5^{\pm 3.0}$	$\mathbf{57.5}^{\pm 2.6}$	$\mathbf{46.4}^{\pm 0.7}$

Table 10: RQ4 – Full per task evaluation results. We report mean success rate (%) with standard deviation across four seeds.

Method	Temporal				Spatial			
	Bin Fill	Pick Xtimes	Swing Xtimes	Stop Cube	Video Unmask	Button Unmask	Video Unmask Swap	Button Unmask Swap
Unfreeze	$\mathbf{53.5}^{\pm 3.6}$	$80.0^{\pm 3.7}$	$84.5^{\pm 5.0}$	$41.0^{\pm 1.7}$	$26.0^{\pm 1.4}$	$36.0^{\pm 1.4}$	$27.0^{\pm 2.2}$	$\mathbf{24.5}^{\pm 2.6}$
0.01	$31.0^{\pm 1.0}$	$87.5^{\pm 2.2}$	$35.5^{\pm 3.0}$	$44.0^{\pm 2.8}$	$40.5^{\pm 1.7}$	$43.0^{\pm 2.2}$	$12.0^{\pm 3.2}$	$21.0^{\pm 1.0}$
Freeze	$28.0^{\pm 2.8}$	$84.5^{\pm 1.7}$	$33.5^{\pm 3.3}$	$39.0^{\pm 3.0}$	$29.0^{\pm 1.7}$	$39.0^{\pm 5.2}$	$22.0^{\pm 1.4}$	$15.0^{\pm 4.1}$
0.03	$35.0^{\pm 3.0}$	$92.0^{\pm 2.4}$	$47.5^{\pm 3.0}$	$43.5^{\pm 5.0}$	$\mathbf{49.5}^{\pm 2.6}$	$39.5^{\pm 3.0}$	$16.5^{\pm 3.0}$	$20.0^{\pm 1.4}$
0.3	$49.5^{\pm 3.0}$	$\mathbf{93.5}^{\pm 1.7}$	$\mathbf{86.0}^{\pm 2.0}$	$\mathbf{47.5}^{\pm 5.0}$	$32.5^{\pm 1.7}$	$34.0^{\pm 3.2}$	$17.5^{\pm 5.4}$	$18.5^{\pm 1.7}$
0.1	$44.0^{\pm 2.0}$	$87.5^{\pm 0.9}$	$72.5^{\pm 5.9}$	$47.0^{\pm 3.3}$	$45.5^{\pm 1.7}$	$\mathbf{45.5}^{\pm 3.0}$	$\mathbf{32.5}^{\pm 3.8}$	$17.0^{\pm 1.7}$

Method	Object				Procedural				AVG
	Pick Highlight	Video Repick	Video Place Button	Video Place Order	Move Cube	Insert Peg	Pattern Lock	Route Stick	
Unfreeze	$17.5^{\pm 2.2}$	$22.0^{\pm 2.0}$	$24.0^{\pm 3.7}$	$22.0^{\pm 4.9}$	$61.5^{\pm 1.7}$	$2.0^{\pm 2.4}$	$10.5^{\pm 2.6}$	$6.5^{\pm 0.9}$	$33.7^{\pm 0.6}$
0.01	$\mathbf{35.0}^{\pm 3.0}$	$33.5^{\pm 5.2}$	$48.0^{\pm 1.4}$	$36.0^{\pm 1.4}$	$69.0^{\pm 5.4}$	$5.5^{\pm 3.8}$	$16.0^{\pm 1.4}$	$50.0^{\pm 3.7}$	$38.0^{\pm 0.4}$
Freeze	$32.5^{\pm 4.1}$	$\mathbf{41.0}^{\pm 3.0}$	$66.0^{\pm 5.1}$	$\mathbf{43.5}^{\pm 1.7}$	$62.0^{\pm 1.4}$	$2.5^{\pm 1.7}$	$22.0^{\pm 1.4}$	$53.0^{\pm 1.7}$	$38.3^{\pm 1.5}$
0.03	$33.0^{\pm 5.4}$	$38.5^{\pm 4.6}$	$61.0^{\pm 1.7}$	$39.5^{\pm 1.7}$	$67.0^{\pm 3.0}$	$6.5^{\pm 3.0}$	$19.0^{\pm 4.4}$	$\mathbf{57.5}^{\pm 4.1}$	$41.6^{\pm 0.4}$
0.3	$27.5^{\pm 2.2}$	$34.5^{\pm 4.1}$	$55.0^{\pm 6.1}$	$26.0^{\pm 3.2}$	$\mathbf{87.5}^{\pm 4.3}$	$7.0^{\pm 2.2}$	$20.0^{\pm 3.2}$	$41.5^{\pm 3.3}$	$42.4^{\pm 0.5}$
0.1	$34.5^{\pm 2.6}$	$39.0^{\pm 3.6}$	$\mathbf{69.0}^{\pm 3.0}$	$35.0^{\pm 2.2}$	$80.0^{\pm 3.2}$	$\mathbf{8.0}^{\pm 2.8}$	$\mathbf{27.5}^{\pm 3.0}$	$\mathbf{57.5}^{\pm 2.6}$	$\mathbf{46.4}^{\pm 0.7}$

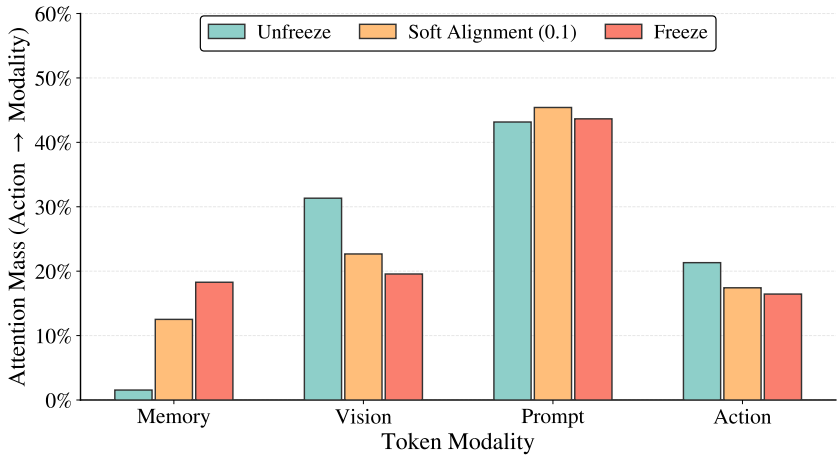

Figure 7: Mean attention mass from action-token queries to tokens of each input modality. We compare *Unfreeze*, which removes the alignment stage by training all components at the full learning rate; *Soft Alignment*, using $\alpha_{\text{align}} = 0.1$; and *Freeze*, using $\alpha_{\text{align}} = 0$.

B.4 RQ4 – Per Task Results

We report the complete evaluation results per task for the RQ4 experiments in Table 10.

B.5 RQ4 – Attention Mass Analysis

Figure 7 provides a qualitative diagnostic of how the alignment schedule affects the interaction between action and memory representations. For each action-token query, we compute the attention mass assigned to the tokens of each modality and average this quantity across the token sequence dimension, attention heads, transformer layers, and multiple simulation episodes. We compare three variants from RQ4: *Unfreeze*, where all components are trained at the full learning rate from the beginning; *Soft Alignment*, using our default $\alpha_{\text{align}} = 0.1$; and *Freeze*, where the pretrained backbone and action expert remain fixed during the alignment stage.

Without staged alignment, action tokens assign almost no attention mass to the memory tokens. In contrast, completely freezing the pretrained components during alignment results in substantially higher attention toward memory. Soft Alignment lies between these two extremes, with action tokens allocating attention to memory while retaining substantially higher task performance than the fully frozen variant. This pattern is consistent with the trade-off observed in RQ4: unrestricted joint fine-tuning can lead to weak interaction with the newly introduced memory pathway, whereas overly restrictive freezing promotes stronger interaction with memory but can impair the pretrained manipulation capabilities. We emphasize that attention mass provides a diagnostic of information routing rather than a direct measure of causal memory utilization.