DUAL LATENT MEMORY IN VISION-LANGUAGE-ACTION MODELS FOR ROBOTIC MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Mainstream Vision-Language-Action (VLA) models predict actions primarily from the current observation under a Markovian assumption, thus struggling with long-horizon, temporally dependent tasks. Existing memory-augmented VLAs either expand the observation window or retrieve history from the memory bank as auxiliary policy-side context. However, they leave memory outside the native latent embedding space of VLA reasoning, preventing historical experience from being fluidly interleaved with multimodal reasoning and action formation. To this end, we introduce **`LaMem-VLA`**, a latent-memory-native framework that reconstructs historical experience into latent memory tokens and directly interweaves them with VLA reasoning. At its core, **`LaMem-VLA`** introduces four coordinated components: ***(i)*** a **curator** that organizes historical experience into two complementary short-term and long-term memory vaults; ***(ii)*** a **seeker** that queries both vaults using the multimodal cognition to retrieve context-relevant evidence; ***(iii)*** a **condenser** that reconstructs the retrieved evidence into compact short-term and long-term latent memory tokens; and ***(iv)*** a **weaver** that injects these memory tokens with the current observation and instruction into one continuous embedding sequence. By representing, retrieving, and consuming historical experience entirely in the same continuous latent space, **`LaMem-VLA`** enables memory to directly participate in VLA reasoning and guide action generation. Experiments on SimplerEnv, LIBERO and MIKASA-Robo demonstrate state-of-the-art success rates of 73.9%, 97.6%, and 51.4%, respectively. In real-world evaluation, **`LaMem-VLA`** ranks best on general and long-horizon suites. The code will be released.

1 INTRODUCTION

Vision-language-action (VLA) models [1, 2, 3, 4] have become a promising paradigm for general robotic manipulation. By combining pretrained vision-language models [5, 6, 7] with policy learning [8, 9, 10] on robotic data [11, 12, 13, 14], they map visual observations and language instructions into executable action chunks. Despite this progress, most existing VLA models [2, 1, 15] implicitly rely on a Markovian assumption, predicting actions primarily from the current observation without considering temporal dependencies. This simplification creates a *temporal short-horizon bias*: VLA models can react to the currently visible state, but struggle to reason over previous state transitions, completed operation steps, and the current phase of a multi-step task, making long-horizon manipulation particularly challenging.

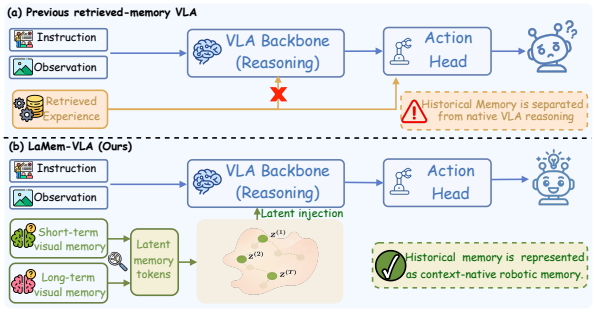


Figure 1: Paradigm comparison of memory-augmented VLA Models. (a) Unlike previous VLA models that store historical experience in an auxiliary memory bank and consume retrieved memory as external policy-side context, (b) **`LaMem-VLA`** treats historical experience as context-native latent memory, which is stored, retrieved, and consumed in the model embedding space.

Recent efforts have sought to alleviate temporal short-horizon bias by augmenting VLA models [8, 16, 17] with historical context or memory mechanisms along two main axes. ***(i)*** One line

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.

of work incorporates short-horizon episode context by concatenating historical frames [18, 19] or extending the input into a video sequence [20, 21, 22]. Although such designs expose recent state changes, they incur computational and memory costs that grow with the context length, while the fixed temporal horizon discards potentially relevant evidence outside the input window. ***(ii)*** Another line of work [23, 24, 25] retrieves past trajectories or relevant historical tokens from an external memory bank to condition downstream action policies. Although these methods demonstrate the value of historical experience for long-horizon manipulation, they still suffer from an architectural limitation: the historical memory is stored outside the model's native token space and consumed as auxiliary policy-side context after the VLA model reasoning. This rigid separation prevents memory from being fluidly interleaved with the internal reasoning where the VLA model jointly perceives the scene, interprets the instruction, and resolves action queries before action formation.

This limitation raises a more fundamental question for memory-dependent VLA models: *whether historical experience can be represented as context-native robotic memory, stored, retrieved, and consumed in the same continuous space where the VLA model already perceives, reasons, and acts?* Latent embedding space [26, 27, 28, 29, 30, 31, 32] offers a natural answer to this question. Modern VLA models already integrate visual observations and language instructions in a continuous token embedding space [33, 2, 1, 15]; therefore, robotic historical memory can be organized as machine-native latent memory tokens that are compatible with the internal reasoning process. Under this formulation, robotic memory becomes part of the model's operating context rather than an auxiliary scaffold attached after multimodal reasoning. Long-horizon robotic manipulation also calls for two complementary forms of historical memory: **short-term memory** is visually dominant, preserving visually grounded evidence from the current episode, such as object locations and subtle state changes; **long-term memory** is semantically dominant, preserving task progress, contextual semantics, and action continuity across longer horizons. Notably, their distinction lies in *provenance and function* rather than in representation form: both are ultimately reconstructed as latent memory that can be consumed by the VLA model in the latent space. This leads to our pivotal research question:

> 🔍 *How can historical experience be organized as complementary visual and semantic memory, and reconstructed into compact latent memory tokens that interweave seamlessly with the VLA reasoning and action generation process?*

To answer this question, we propose `LaMem-VLA`, a novel native latent memory framework for robotic manipulation, which explicitly organizes robotic history into two complementary memory vaults, and weaves dual-source memory into the model reasoning for memory-augmented action generation. At its core, `LaMem-VLA` closes the loop between latent memory weaving and action reasoning through four coordinated modules: First, a **latent memory curator** factorizes past robotic experience into two complementary vaults: a short-term memory vault for recent visual evidence and a long-term memory vault for semantic and action-continuity evidence. Second, during the action reasoning process, a **latent memory seeker** builds a context-aware query from the current multimodal cognition state (*i.e.*, the visual and instruction tokens), and uses it to retrieve task-relevant historical evidence from dual memory vaults, grounding the present decision in past perceptual evidence and long-horizon semantic-action continuity. Third, a **latent memory condenser** reconstructs the retrieved evidence as fixed-length short-term and long-term latent memory tokens compatible with the VLA embedding space. Finally, a **latent memory weaver** inserts these condensed memory tokens into the action reasoning sequence before action query tokens are resolved, allowing historical memory to participate in the same latent reasoning process as the current image, instruction, and action queries. The resulting memory-grounded action queries condition a diffusion-based action expert [8, 34] to generate temporally aware robotic action sequences.

We conduct comprehensive evaluations on three simulation benchmarks (*i.e.*, LIBERO [35], SimplerEnv-Bridge [36], and MIKASA-Robo [37]) and six real-world manipulation tasks. On LIBERO, SimplerEnv-Bridge, and MIKASA-Robo, `LaMem-VLA` achieves average success rates of **97.6**%, **73.9**%, and **51.4**%, outperforming MemoryVLA [23] by **1.1**, **2.0**, and **10.2** points, respectively. In real-world evaluations, `LaMem-VLA` achieves **77.3**% and **61.3**% average success on the Generalization and Long-horizon task suites, respectively, outperforming the strongest baseline by **15.3** points on the latter. These results demonstrate the effectiveness of integrating complementary latent memory into VLA reasoning, particularly when action generation depends on historical cues.

In summary, our main contributions are as follows: **First**, we introduce a **new paradigm** for robotic memory in VLA models: historical experience is treated as context-native latent memory, which is

stored, retrieved, and consumed in the model embedding space, supporting scene perception, instruction understanding, and action-intent formation within the same latent reasoning process. **Second**, we propose `LaMem-VLA`, a **dual latent memory framework** for robotic manipulation, which explicitly organizes robotic history into two complementary memory vaults, and weaves dual-source memory into the model reasoning for action generation. **Third**, we design a **latent memory condensing mechanism**, which transforms retrieved historical evidence into fixed-length latent short-term and long-term tokens that are compatible with the VLA model embedding space.

2 Related Work

Vision-Language-Action (VLA) Models. Driven by advances in pretrained vision-language models (VLMs), VLA models [38, 39, 1, 40] have demonstrated promising performance in robotic control via mapping visual observations and language instructions to robot actions. According to their action policy design, VLA models can be broadly classified into two paradigms: single-stream architecture and hierarchical architectures. **(i)** Single-stream models [41, 42, 43, 44] directly discretize continuous actions [33, 2] within a unified vision-language backbone, and autoregressively predict action tokens. To meet the high control-frequency requirements in robotic manipulation, many works focus on inference efficiency optimization [45, 46, 47, 48]. **(ii)** Hierarchical models [3, 49, 50] typically employ VLMs for high-level reasoning and planning, and adopt a separate generative policy, such as diffusion-based [1, 15] and flow-matching-based [51, 9] policies, to synthesize smooth and high-quality action trajectories. Despite these advances, these methods still predict actions primarily from the current observation and instruction without leveraging extended historical context, limiting their robustness in long-horizon manipulation tasks. In contrast, our work aims to equip VLA models with explicit progress awareness via memory mechanisms, addressing memory-dependent tasks that current approaches fail to handle.

Memory Mechanisms for Robotic Control. Mainstream VLA policies are formulated under a Markovian assumption, predicting actions primarily from the current observation and therefore lacking explicit awareness of task progress. Existing attempts to incorporate historical context can be broadly categorized into three groups. **(i)** *Temporal-context expansion:* This line of work exposes the policy to extended observations by interleaving historical frames with language tokens [18, 19] or aggregating multi-frame features [20, 21, 22] within a predefined context window. However, their computational and memory costs grow with context length, while relevant evidence outside the fixed window is discarded. **(ii)** *Sparse history abstraction:* Another paradigm compresses past interactions into lightweight proxy representations, such as recurrent latent states [52, 53], motion-centric cues [54], action summaries [55], or a sparse set of representative observations [56]. However, such abstraction may discard fine-grained perceptual details and high-level semantic context. **(iii)** *External memory conditioning:* A third direction [23, 24, 25] stores historical observations and action tokens in an auxiliary memory bank, and retrieves task-relevant evidence for action prediction. Since memory access is driven primarily by the current query, retrieval may become unreliable under weak visual cues or similar distractors. More fundamentally, the retrieved memory is typically consumed as auxiliary policy-side context rather than integrated into the VLA model's native token-level reasoning process. In contrast, `LaMem-VLA` maintains complementary short-term visual and long-term semantic memory vaults, from which decision-relevant evidence is retrieved and distilled into compact latent tokens. These tokens are directly interwoven with current visual, language, and action tokens, allowing perceptual evidence and long-range task progress to jointly shape action formation.

3 Method

3.1 Problem Formulation

We formulate the robotic manipulation task in VLA models as a language-conditioned Markovian decision-making problem. At each timestep t, the VLA policy [46, 2, 15], $\mathbf{\Pi}_\theta$ takes a natural-language instruction $\boldsymbol{I}$ and the current visual observation $\boldsymbol{o}_t$ as input, and predicts a chunk of future actions for executing the specified task:

$$\boldsymbol{a}_{t:t+H-1} = \mathbf{\Pi}_\theta(\boldsymbol{o}_t, \boldsymbol{I}), \tag{1}$$

where H denotes the action horizon, and each action $a_t \in \mathbb{R}^7$ is a 7-DoF end-effector control vector, consisting of 3-DoF relative translation, 3-DoF relative rotation, and a 1-DoF gripper state.

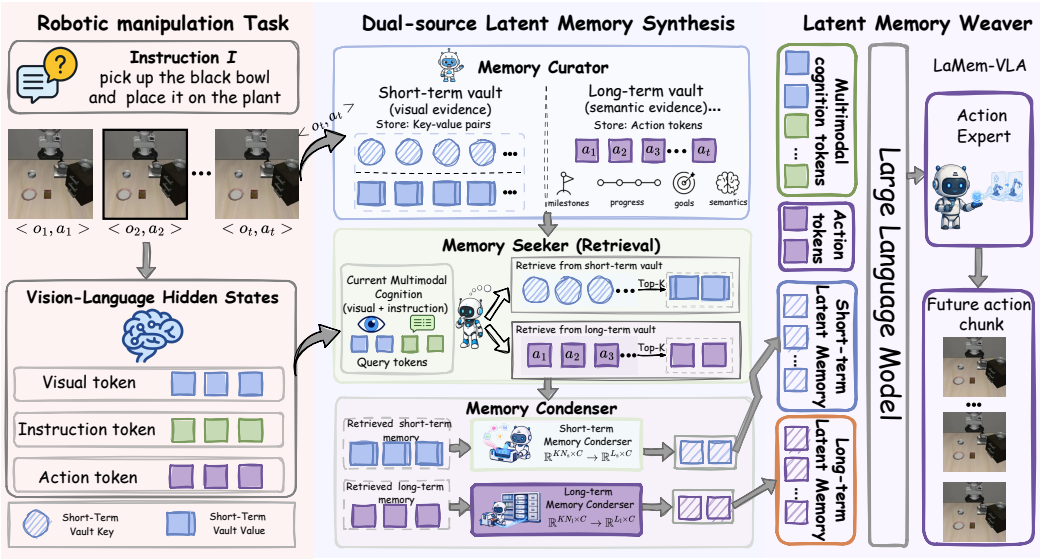


Figure 2: **The Framework of `LaMem-VLA`.** The memory curator (§3.3) organizes visual tokens and action tokens into dual memory vaults, and the memory seeker (§3.4) then retrieves task-relevant evidence from dual memory vaults based on visual and instruction tokens. This retrieved evidence is compressed into fixed-length latent memory tokens by the memory condenser (§3.4). Finally, the memory weaver (§3.5) injects these latent memory tokens into the reasoning sequence before the action queries are resolved, producing memory-grounded action tokens that guide the action expert to generate future action chunks.

3.2 `LaMem-VLA` Model Architecture

Motivation. The Markovian formulation above in VLA models makes action prediction primarily depend on the instantaneous observation and instruction. Although effective for short-horizon behaviors, this paradigm can induce a temporal short-horizon bias in long-horizon manipulation tasks. Existing works attempt to augment VLA or visuomotor policies with memory through incorporating explicit episode context into the input [57, 58, 20] or externalized memory bank retrieval [23, 24]. However, they store the historical memory outside the model's native token space and consume it as auxiliary policy-side context for action decoding, preventing memory from being interleaved with the internal reasoning process that jointly perceives, reasons, and forms action intent. Thus, our `LaMem-VLA` reformulates robotic memory as context-native latent memory to bridge this gap.

Overview. `LaMem-VLA` is an end-to-end native latent memory framework for robotic manipulation that directly weaves dual-source historical experience into VLA reasoning to refine action generation, as shown in Fig. 2. `LaMem-VLA` closes the loop between latent memory reconstruction and action reasoning through four coordinated modules. At each timestep, the vision-language backbone embeds current visual observation and instruction into the visual and instruction tokens, and learnable action queries are appended to the token sequence to obtain manipulation-relevant latent action representations. First, a **latent memory curator** dynamically establishes and updates two complementary memory vaults: the short-term memory vault $\mathcal{M}^{\text{short}}$ stores visual tokens that preserve recent perceptual evidence, while the long-term memory vault $\mathcal{M}^{\text{long}}$ stores action tokens that preserve semantic and action-continuity evidence across longer horizons. Second, a **latent memory seeker** constructs a context-aware query $\boldsymbol{Q}_t^{\text{con}}$ from the current multimodal cognition representation and uses it to retrieve decision-relevant history from the two vaults. Third, a **latent memory condenser** distills the retrieved raw memory content into compact latent short-term memory tokens $\boldsymbol{M}^{\text{short}}$ and latent long-term memory tokens $\boldsymbol{M}^{\text{long}}$, respectively. Finally, a **latent memory weaver** interleaves these tokens with the current image, instruction, and action queries, whose memory-grounded representations condition a diffusion-based action expert [34] to generate action sequences.

3.3 Latent Memory Curator

The latent memory curator maintains the historical evidence that will later be retrieved, condensed, and woven into the VLA reasoning sequence. `LaMem-VLA` adopts a 7B-parameter Prismatic

vision-language model [59] as the backbone, which is further pretrained on the large-scale cross-embodiment real robotic dataset Open-X Embodiment [11]. For the current RGB observation, the backbone first extracts visual tokens with its vision encoder [60, 61] and projects them into the language embedding space to obtain final visual tokens $\boldsymbol{X}_t \in \mathbb{R}^{N_\mathrm{i} \times C}$, where N_i is the sequence length and C is the hidden dimension. These visual tokens are concatenated with the tokenized instruction and processed by LLaMA-7B [62]. We append learnable action queries $\boldsymbol{Q}^{\text{action}} \in \mathbb{R}^{N_\mathrm{a} \times C}$ to the sequence, and use their output hidden states $\boldsymbol{H}^{\text{action}}$ as compact action representations for downstream action prediction. The curator factorizes the historical evidence into two memory vaults: a short-term memory vault $\mathcal{M}^{\text{short}}$ and a long-term memory vault $\mathcal{M}^{\text{long}}$. Both vaults are initialized as empty at the beginning of each episode. Each vault has a maximum capacity of L memory units.

Short-term Memory Vault. The short-term memory vault $\mathcal{M}^{\text{short}}$ stores visual perceptual evidence from the current episode. Each short-term memory unit is represented as a key-value pair $\boldsymbol{m}_\mathrm{s}^i = (\boldsymbol{k}_\mathrm{s}, \boldsymbol{v}_\mathrm{s})$: the key $\boldsymbol{k}_\mathrm{s}$ provides a concise retrieval summary of visual evidence, while the value $\boldsymbol{v}_\mathrm{s}$ stores the latent short-term memory content. Specifically, at each timestep, a learnable compression module distills the current visual tokens $\boldsymbol{X}_t$ into a compact set of short-term memory tokens, and their mean-pooled representation is used as the retrieval key:

$$\boldsymbol{v}_\mathrm{s} = \mathcal{C}_\mathrm{s}(\boldsymbol{X}_t) \in \mathbb{R}^{N_\mathrm{s} \times C}, \quad \boldsymbol{k}_\mathrm{s} = \mathrm{MeanPool}(\boldsymbol{v}_\mathrm{s}) \in \mathbb{R}^{C}, \tag{2}$$

where $\mathcal{C}_\mathrm{s}$ is an SE-bottleneck compression module [63], and the unit $\boldsymbol{m}_\mathrm{s}$ is then appended to $\mathcal{M}^{\text{short}}$.

Long-term Memory Vault. The long-term memory vault $\mathcal{M}^{\text{long}}$ stores the action hidden states that track task progress and action continuity across long horizons. At each timestep, the curator directly writes the action hidden states $\boldsymbol{H}^{\text{action}}$ into the vault as one long-term memory unit $\boldsymbol{m}_\mathrm{l}$.

Memory Vault Updating Strategy. After a new memory unit is written into its corresponding vault, the memory curator applies a compression strategy only when the number of stored units exceeds the capacity L. For the short-term visual stream, let $\mathcal{M}^{\text{short}} = \{\boldsymbol{m}_\mathrm{s}^i = (\boldsymbol{k}_\mathrm{s}^i, \boldsymbol{v}_\mathrm{s}^i)\}_{i=1}^{n_\mathrm{s}}$ after insertion. If $n_\mathrm{s} > L$, we compute the cosine similarity between temporally adjacent keys and select the most redundant adjacent pair:

$$i_\mathrm{s}^* = \arg\max_{1 \le i < n_\mathrm{s}} \cos(\boldsymbol{k}_\mathrm{s}^i, \boldsymbol{k}_\mathrm{s}^{i+1}). \tag{3}$$

The selected pair is consolidated by averaging both its key and value tokens:

$$\tilde{\boldsymbol{k}}_\mathrm{s} = \frac{1}{2}(\boldsymbol{k}_\mathrm{s}^{i_\mathrm{s}^*} + \boldsymbol{k}_\mathrm{s}^{i_\mathrm{s}^*+1}), \qquad \tilde{\boldsymbol{v}}_\mathrm{s} = \frac{1}{2}(\boldsymbol{v}_\mathrm{s}^{i_\mathrm{s}^*} + \boldsymbol{v}_\mathrm{s}^{i_\mathrm{s}^*+1}), \qquad \tilde{\boldsymbol{m}}_\mathrm{s} = (\tilde{\boldsymbol{k}}_\mathrm{s}, \tilde{\boldsymbol{v}}_\mathrm{s}). \tag{4}$$

The two adjacent units are replaced by $\tilde{\boldsymbol{m}}_\mathrm{s}$, reducing redundancy while preserving their shared visual evidence. For the long-term vault, adjacent units are ranked by the similarity of their mean-pooled action representations and consolidated by averaging. More details are provided in Appendix §A.1.

3.4 Learning to Synthesize Latent Memory with Dual-source Vault

`LaMem-VLA` does not expose memory vaults to the action policy as raw auxiliary context. Instead, it treats the two vaults as latent evidence substrates: given the current multimodal cognition representation, `LaMem-VLA` first retrieves relevant evidence from the two vaults and then reconstructs it into compact memory tokens that are native to the VLA embedding space.

Latent Memory Seeker. The latent memory seeker retrieves evidence from the memory vaults according to the current multimodal context rather than the visual observation or language instruction alone. Given the current visual tokens $\boldsymbol{X}_t$ and instruction tokens $\boldsymbol{I}$, the VLA backbone produces a context-aware query $\boldsymbol{Q}_t^{\text{con}}$ that encodes the current visual-linguistic cognition. The seeker then appends learnable query slots $\boldsymbol{Q}^{\text{init}} \in \mathbb{R}^{K_\mathrm{q} \times C}$ to $\boldsymbol{Q}_t^{\text{con}}$ and only updates the query slots:

$$\boldsymbol{Q}_t = \mathcal{B}([\boldsymbol{Q}_t^{\text{con}}; \boldsymbol{Q}^{\text{init}}])[-K_\mathrm{q}:] \in \mathbb{R}^{K_\mathrm{q} \times C}, \tag{5}$$

where $\mathcal{B}$ is a transformer-based query builder (more details in Appendix) and $\boldsymbol{Q}_t$ serves as the shared memory hook for both vaults. We apply masked attention inside $\mathcal{B}$ such that the appended query slots read from $\boldsymbol{Q}_t^{\text{con}}$, while the original multimodal hidden states are not perturbed by the query slots. The mean-pooled query $\boldsymbol{q}_t = \mathrm{MeanPool}(\boldsymbol{Q}_t) \in \mathbb{R}^C$ is used as the global retrieval vector.

The memory seeker then uses $\boldsymbol{q}_t$ to retrieve context-relevant units from the short-term vault $\mathcal{M}^{\text{short}} = \{(\boldsymbol{k}_\mathrm{s}^i, \boldsymbol{v}_\mathrm{s}^i)\}_{i=1}^{|\mathcal{M}^{\text{short}}|}$ by cosine similarity:

$$\mathcal{I}_\mathrm{s} = \text{Top–}K\left(\{\cos(\boldsymbol{q}_t, \boldsymbol{k}_\mathrm{s}^i)\}_{i=1}^{|\mathcal{M}^{\text{short}}|}\right), \quad \boldsymbol{Z}^{\text{short}} = \mathrm{Concat}_{i \in \mathcal{I}_\mathrm{s}}(\boldsymbol{v}_\mathrm{s}^i) \in \mathbb{R}^{K N_\mathrm{s} \times C}, \tag{6}$$

Table 1: **Quantitative comparison results on SimplerEnv-Bridge [36]** (§4.2) with WidowX robot.

Method	Publication	Spoon on Towel	Carrot on Plate	Stack Cube	Eggplant in Basket	Avg. Success
RT-1-X [11]	ICRA2024	0.0	4.2	0.0	0.0	1.1
OpenVLA [2]	CoRL2025	4.2	0.0	0.0	12.5	4.2
TraceVLA [54]	ICLR2025	12.5	16.6	16.6	65.0	27.7
SpatialVLA [64]	RSS2025	16.7	25.0	29.2	100.0	42.7
CogACT [15]	ArXiv2024	58.3	45.8	29.2	95.8	57.3
π_0 [1]	RSS2025	83.8	52.5	52.5	87.9	69.2
DreamVLA [65]	NeurIPS2025	45.8	45.8	25.0	87.5	51.0
ThinkAct [66]	NeurIPS2025	58.3	37.5	8.7	70.8	43.8
CronusVLA [20]	AAAI2026	66.7	54.2	20.8	100.0	60.4
MemoryVLA [23]	ICLR2026	75.0	75.0	37.5	100.0	71.9
SemanticVLA [67]	CVPR2026	83.6	54.5	40.3	81.8	65.1
`LaMem-VLA` (Ours)	–	83.3	75.0	41.7	95.8	**73.9**

where K is the number of retrieved short-term memory units. For long-term memory retrieval, the seeker similarly ranks the units in $\mathcal{M}^{\text{long}} = \{\boldsymbol{m}_{\text{l}}^i\}_{i=1}^{|\mathcal{M}^{\text{long}}|}$ via comparing the mean-pooled long-term units with the mean-pooled query $\boldsymbol{q}_t$. It then selects the Top–K long-term memory units and stacks them into $\boldsymbol{Z}^{\text{long}} \in \mathbb{R}^{KN_{\text{l}} \times C}$. The retrieved sets $\boldsymbol{Z}^{\text{short}}$ and $\boldsymbol{Z}^{\text{long}}$ are passed to the memory condenser below as short-term visual evidence and long-term semantic evidence, respectively, instead of being inserted verbatim into the VLA sequence.

Latent Memory Condenser. Directly inserting the retrieved evidence sequences into the VLA sequence would expand the reasoning context and introduce redundant historical tokens, so the latent memory condenser reconstructs them as fixed-length tokens in the VLA embedding space. Specifically, we introduce learnable short-term visual memory slots $\boldsymbol{T}_{\text{s}} \in \mathbb{R}^{L_{\text{s}} \times C}$ and long-term semantic memory slots $\boldsymbol{T}_{\text{l}} \in \mathbb{R}^{L_{\text{l}} \times C}$, and update them with lightweight memory formers conditioned on the context query tokens $\boldsymbol{Q}_t \in \mathbb{R}^{K_{\text{q}} \times C}$ and the retrieved evidence:

$$\boldsymbol{M}^{\text{short}} = \mathcal{F}_{\text{v}}([\boldsymbol{Q}_t; \boldsymbol{Z}^{\text{short}}; \boldsymbol{T}_{\text{s}}])[-L_{\text{s}}:], \quad \boldsymbol{M}^{\text{long}} = \mathcal{F}_{\text{c}}([\boldsymbol{Q}_t; \boldsymbol{Z}^{\text{long}}; \boldsymbol{T}_{\text{l}}])[-L_{\text{l}}:], \tag{7}$$

where $\mathcal{F}_{\text{v}}$ and $\mathcal{F}_{\text{c}}$ denote lightweight transformer-style memory formers similar to $\mathcal{B}$ for the short-term visual and long-term semantic memory, respectively. The resulting $\boldsymbol{M}^{\text{short}}$ and $\boldsymbol{M}^{\text{long}}$ are query-conditioned latent short-term and long-term memory tokens in the same C-dimensional embedding space used by VLA reasoning. This fixed-length property makes the injected memory independent of the retrieval size while preserving evidence from the two vaults.

3.5 Interweaving Memory into Action Reasoning in Latent Space

Latent Memory Weaver for Guidance Generation. Rather than passing the condensed memory tokens to a policy head as external conditioning alone, the latent memory weaver injects the synthesized memory into the VLA reasoning sequence before action queries are resolved. Although the two types of latent memory differ in provenance, they share the same latent token interface. Given the condensed short-term memory tokens $\boldsymbol{M}^{\text{short}}$, long-term memory tokens $\boldsymbol{M}^{\text{long}}$, the weaver constructs the memory-augmented VLA input sequence $\boldsymbol{S}_t$:

$$\boldsymbol{S}_t = [\boldsymbol{M}^{\text{short}} + \mathbf{1}_{L_{\text{s}}} \boldsymbol{b}_{\text{s}}^\top; \boldsymbol{M}^{\text{long}} + \mathbf{1}_{L_{\text{l}}} \boldsymbol{b}_{\text{l}}^\top; \boldsymbol{X}_t; \boldsymbol{I}; \boldsymbol{Q}^{\text{action}}], \quad \boldsymbol{Z}^{\text{action}} = \text{VLM}(\boldsymbol{S}_t)[-N_{\text{a}}:], \tag{8}$$

where $\boldsymbol{b}_{\text{s}}, \boldsymbol{b}_{\text{l}} \in \mathbb{R}^C$ are learnable source embeddings for the two memory streams, and $\mathbf{1}_{L_{\text{s}}}, \mathbf{1}_{L_{\text{l}}}$ are all-one column vectors that broadcast them over the short-term and long-term memory tokens. Because the two sources of memory tokens are part of the model input sequence, they participate in self-attention with the current observation, language instruction, and action queries.

Diffusion-based Action Expert. After the vision-language backbone produces the memory-grounded action tokens $\boldsymbol{Z}^{\text{action}}$, the diffusion-based action expert decodes them into a continuous action chunk. Following the diffusion policy [8], we formulate action generation as conditional denoising over an action chunk. Let $\boldsymbol{a}_{t:t+H-1}^0$ denote the clean expert action chunk and $\boldsymbol{a}_{t:t+H-1}^n$ denote its noisy version at diffusion step n. At each denoising step, the noisy action tokens are injected with the diffusion timestep embedding and conditioned on action tokens $\boldsymbol{Z}^{\text{action}}$.

The diffusion expert ϵ_θ is trained with mean squared error (MSE) loss to predict the injected noise under these action and memory conditions:

$$\mathcal{L}_{\text{action}} = \mathbb{E}_{n,\epsilon}\left[\left\|\epsilon - \epsilon_\theta(\boldsymbol{a}_{t:t+H-1}^n, n, \boldsymbol{Z}^{\text{action}})\right\|_2^2\right]. \tag{9}$$

Table 2: **Quantitative comparison results on MIKASA-Robo [37] (§4.3)** with the Franka robot.

Method	Publication	ShellGame Touch	Intercept Medium	Remb. Color-3	Remb. Color-5	Remb. Color-9	Avg. Success
CronusVLA [20]	AAAI2026	32	5	31	13	9	18.0
SpatialVLA [64]	RSS2025	23	27	27	17	11	21.0
OpenVLA-OFT [46]	ArXiv2025	47	14	**59**	16	6	28.4
π_0 [1]	RSS2025	33	42	35	22	15	29.4
Octo [10]	RSS2024	46	39	45	17	11	31.6
MemoryVLA [23]	ICLR2026	88	24	44	30	20	41.2
LaMem-VLA (Ours)	–	**94**	**43**	58	**37**	**25**	**51.4**

Table 3: **Quantitative comparison results on LIBERO [35] (§4.4) with Franka robot.** Success rates (%) are reported across five suites. * indicates methods using additional proprioceptive and wrist-camera inputs. For methods without LIBERO-90 results, we report the average over the first four suites.

Method	Publication	Spatial	Object	Goal	Long-10	Long-90	Avg. Success
Diffusion Policy [8]	IJRR2025	78.3	92.5	68.3	50.5	–	72.4
Octo [10]	RSS2024	78.9	85.7	84.6	51.1	–	75.1
UniACT [69]	CVPR2025	77.0	87.0	77.0	70.0	73.0	76.8
SpatialVLA [64]	RSS2025	88.2	89.9	78.6	55.5	46.2	71.7
OpenVLA [2]	CoRL2025	84.7	88.4	79.2	53.7	73.5	75.9
CoT-VLA [41]	CVPR2025	87.5	91.6	87.6	69.0	–	83.9
π_0-FAST* [70]	ArXiv2025	96.4	96.8	88.6	60.2	83.1	85.0
CogACT [15]	ArXiv2024	97.2	98.0	90.2	88.8	92.1	93.2
π_0^* [1]	RSS2025	96.8	98.8	95.8	85.2	–	94.2
ThinkAct [66]	NeurIPS2025	88.3	91.4	87.1	70.9	–	84.4
MemoryVLA [23]	ICLR2026	98.4	98.4	96.4	93.4	95.6	96.5
Fast-ThinkAct [71]	CVPR2026	92.0	97.2	90.2	79.4	–	89.7
SemanticVLA [67]	CVPR2026	98.0	98.6	96.8	94.4	–	97.0
LARA [72]	ICML2026	96.5	97.5	96.0	92.5	–	95.6
LaMem-VLA (Ours)	–	98.8	99.0	97.2	95.8	97.0	**97.6**

During inference, DDIM sampling [68] iteratively denoises the action chunk under the same conditions, producing history-aware continuous 7-DoF control actions.

4 Experiment

4.1 Implementation Details

We instantiate **LaMem-VLA** with a 7B-parameter Prismatic VLM backbone and a diffusion action expert with approximately 300M parameters. We train the model on 8 NVIDIA H800 GPUs using PyTorch FSDP. Each GPU processes 32 samples, resulting in a global batch size of 256, and the learning rate is set to 2×10^{-5}. Both the memory vaults keep a maximum capacity of $L = 16$ units. The seeker retrieves the top $K = 8$ units from each vault, and the condenser reconstructs the retrieved evidence into $L_s = 8$ short-term memory tokens and $L_l = 4$ long-term memory tokens. The query builder $\mathcal{B}$ and the two memory formers $\mathcal{F}_v$ and $\mathcal{F}_c$ are implemented as two-layer transformer blocks with masked attention (more details in Appendix A.2). During inference, the diffusion action expert generates actions with DDIM [68] sampling using 10 denoising steps.

4.2 Simulated Evaluation on SimplerEnv-Bridge

Training and Evaluation Protocol. SimplerEnv-Bridge [36] evaluates real-to-sim generalization of robot manipulation policies trained on real-world data. **LaMem-VLA** is trained on the Bridge v2 dataset [13] for 50k optimization steps, with validation performed every 2.5k steps. The final results are reported using the checkpoint that achieves the best validation performance. For evaluation, each task is tested over 24 trials, and we report the average success rate across trials.

Evaluation Results. As shown in Table 1, **LaMem-VLA** achieves the highest average success rate of **73.9**% on SimplerEnv-Bridge, outperforming CogACT [15] by **16.6** points and surpassing recent VLAs such as π_0 [1] and SemanticVLA [67]. These results indicate that injecting dual latent memory tokens into the reasoning sequence provides effective historical context for action prediction.

Table 4: **Detailed analysis of dual-source latent memory** on SimplerEnv [36] and LIBERO [35] (§4.6).

Method	SimplerEnv	Long-90
w/o Dual Memory	57.3	92.1
w/o Short-term Memory	65.6	95.4
w/o Long-term Memory	64.6	94.8
LaMem-VLA (Ours)	**73.9**	**97.0**

Table 5: **Ablation study of latent-native memory integration** on SimplerEnv [36] and LIBERO [35] (§4.6).

Method	SimplerEnv	Long-90
BASELINE	57.3	92.1
Policy-side Memory	71.9	94.8
Raw Retrieval Conditioning	69.8	95.1
LaMem-VLA (Ours)	**73.9**	**97.0**

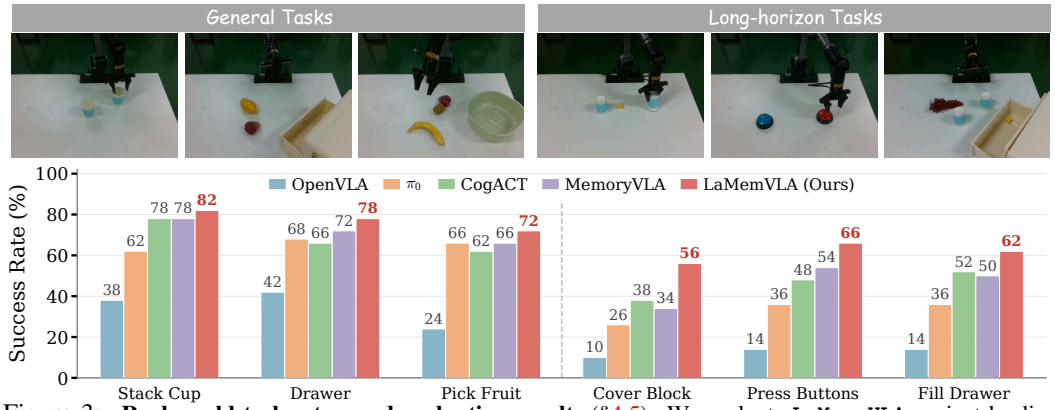


Figure 3: **Real-world task setup and evaluation results** (§4.5). We evaluate **LaMem-VLA** against leading models [2, 1, 15, 23] on the Generalization and Long-horizon task suites.

4.3 Simulated Evaluation on Mikasa-Robo

Training and Evaluation Protocol. We evaluate long-horizon temporal tasks on Mikasa-Robo [37] and compare it with representative VLA models. Following the standard protocol, we use 250 demonstrations per task, 128 × 128 RGB observations, end-effector control, and 100 evaluation episodes per task. The five tasks are jointly trained for 20K steps, with validation performed every 1K steps, and results are reported using the best validation checkpoint.

Evaluation Results. **LaMem-VLA** consistently achieves the highest performance on four of the five tasks, with a **10.2**% improvement in average success rate over the previous state of the art, including a **7.0**% gain on RememberColor-5. The results are shown in Table 2.

4.4 Simulated Evaluation on LIBERO

Training and Evaluation Protocol. We evaluate **LaMem-VLA** on the LIBERO [35] benchmark using a Franka robot across five suites: spatial awareness (Spatial), object manipulation (Object), goal completion (Goal), and long-horizon reasoning (Long-10 and Long-90). The first four suites comprise 10 tasks each, while Long-90 contains 90 tasks. Following the OpenVLA protocol [2], we use 50 demonstrations per task. We train separate models for Spatial, Object, and Goal for 20k optimization steps, and jointly train on Long-10 and Long-90 for 40k steps. We report success rates for each suite and the overall average over 50 rollouts per task.

Evaluation Results. As shown in Table 3, **LaMem-VLA** achieves the highest average success rate of **97.6**% across the five LIBERO suites, outperforming MemoryVLA [23] and CogACT [15] by **1.1** and **4.4** points, respectively. It also surpasses π_0 [1] by **3.5** points over the first four suites, without using the additional proprioceptive and wrist-camera inputs. Notably, **LaMem-VLA** obtains **95.8**% on Long-10 and **97.0**% on Long-90, exceeding MemoryVLA by **2.4** and **1.4** points.

4.5 Real-world Evaluation

Training and Evaluation Protocol. To validate the effectiveness of **LaMem-VLA** in real-world robotic manipulation, we conduct experiments on the AgileX Piper, a 6-DoF single-arm manipulator equipped with a parallel-jaw gripper and a single exocentric Intel RealSense D435i RGB camera. Our evaluation protocol encompasses two task suites: *General Tasks* and *Long-horizon Tasks*. For the former, we collect 200 expert demonstrations per task and conduct 50 evaluation rollouts. For

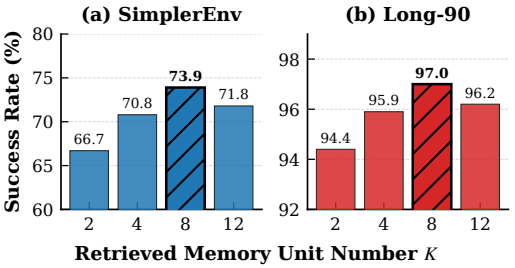


Figure 4: **Ablation of retrieved memory unit number** K on SimplerEnv [36] and Long-90 [35] (§4.6).

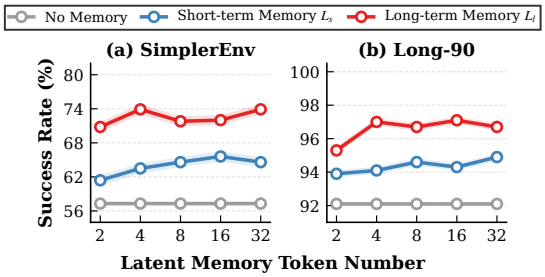


Figure 5: **Ablation study of the latent memory token number** on SimplerEnv [36] and Long-90 [35] (§4.6).

the latter, we utilize 300 expert demonstrations and conduct 50 rollouts per task. In all evaluations, object positions are randomly initialized. More details are provided in Appendix C.4.

Results on Real-World. As illustrated in Fig. 3, `LaMem-VLA` achieves average success rates of 77.3% and 61.3% on General and Long-horizon Tasks, respectively. On the general tasks, it exceeds the strongest baseline on every task, with a gain of up to 6 points on Pick Fruit. On the long-horizon tasks, where correct decisions depend on earlier events, `LaMem-VLA` outperforms the strongest baseline by 15.3 points. These results demonstrate strong real-world competence of `LaMem-VLA`.

4.6 Ablation Study

Effectiveness of Dual-source Latent Memory. We evaluate the contribution of each memory source by selectively removing it from the latent VLA input sequence. We consider four settings: the full `LaMem-VLA`, *w/o Short-term Memory*, *w/o Long-term Memory*, and *w/o Dual-source Memory*. Table 4 shows that the full dual-memory design achieves the best performance on both benchmarks, reaching **73.9**% on SimplerEnv [36] and **97.0**% on Long-90 [35]. Removing both memory streams causes the largest degradation, dropping performance to 57.3% and 92.1%, respectively. The degradation caused by removing either stream verifies their complementary contributions.

Latent-native Memory *vs.* Policy-side Condition. We isolate the effect of memory consumption path in Table 5, where BASELINE denotes the memory-free action policy. The baseline reaches only 57.3% on SimplerEnv [36] and 92.1% on Long-90 [35], while adding memory as an external policy-side condition [23] improves the baseline performance. Directly feeding raw retrieved evidence to the action policy also improves over the baseline, but remains below the full model. Our latent-native `LaMem-VLA` achieves the best results of **73.9**% and **97.0**%. These results confirm that the gain comes from latent-native memory integration rather than merely adding memory as external context.

The Number of Retrieved Memory Units K. As shown in Fig. 4, performance improves as K increases from 2 to 8, reaching **73.9**% on SimplerEnv [36] and **97.0**% on Long-90 [35], but declines slightly at $K = 12$. This indicates that a moderate retrieval budget balances historical coverage against redundant evidence; therefore, we set $K = 8$ by default.

The Number of Latent Memory Tokens L_s and L_l. We study the effects of short- and long-term memory capacity in Fig. 5. Increasing L_s from 2 to 16 improves SimplerEnv [36] success rate from 61.4% to 65.6%, while the gain saturates at 32. With $L_s = 8$, increasing L_l further improves performance, reaching 73.9% on SimplerEnv and 97.0% on Long-90 [35]. Considering the additional self-attention cost of larger memory budgets, we adopt $(L_s, L_l) = (8, 4)$ as the default.

5 Conclusion

In this paper, we propose `LaMem-VLA`, a dual latent memory framework for reducing the temporal short-horizon bias of vision-language-action models. The core idea is to make robotic history part of the model's native latent context, rather than an external condition attached after multimodal reasoning. `LaMem-VLA` organizes historical experience into complementary short-term visual and long-term semantic memory, and reconstructs retrieved evidence as compact latent tokens that directly participate in VLA reasoning and action generation. Extensive evaluations across three simulation benchmarks and real-world manipulation tasks demonstrate its effectiveness, highlighting context-native latent memory as a promising direction for temporally aware VLA systems.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language editing, improve the clarity and presentation of the manuscript, support the retrieval and discovery of relevant literature, and assist with parts of the coding process. All AI-assisted content, including generated code, was carefully reviewed and verified by the authors. All scientific ideas, methodological designs, experimental results, and conclusions were independently developed and validated by the authors. We take full responsibility for the final content of this work.

REFERENCES

[1] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024. 1, 2, 3, 6, 7, 8, 19

[2] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, et al. Openvla: An open-source vision-language-action model. In *CoRL*, pages 2679–2713, 2025. 1, 2, 3, 6, 7, 8, 18

[3] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. In *ICLR*, volume 2025, pages 29982–30009, 2025. 1, 3

[4] Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. In *ICML*, pages 61229–61245, 2024. 1

[5] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024. 1

[6] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023. 1

[7] Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, et al. On scaling up a multilingual vision and language model. In *CVPR*, pages 14432–14444, 2024. 1

[8] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025. 1, 2, 6, 7

[9] Qinglun Zhang, Zhen Liu, Haoqiang Fan, Guanghui Liu, Bing Zeng, and Shuaicheng Liu. Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation. In *AAAI*, volume 39, pages 14754–14762, 2025. 1, 3

[10] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *RSS*, 2024. 1, 7

[11] Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *ICRA*, pages 6892–6903, 2024. 1, 5, 6

[12] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024. 1

[13] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *CoRL*, pages 1723–1736, 2023. 1, 7

[14] Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025. 1

[15] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024. 1, 2, 3, 6, 7, 8, 19, 20, 21

[16] Hang Li, Qian Feng, Zhi Zheng, Jianxiang Feng, Zhaopeng Chen, and Alois Knoll. Language-guided object-centric diffusion policy for generalizable and collision-aware robotic manipulation. *arXiv preprint arXiv:2407.00451*, 2024. 1

[17] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *RSS*, 2024. 1

[18] Xinghang Li, Peiyan Li, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Tao Kong, Hanbo Zhang, and Huaping Liu. Towards generalist robot policies: What matters in building vision-language-action models. *arXiv preprint arXiv:2412.14058*, 2024. 2, 3, 22

[19] Cunxin Fan, Xiaosong Jia, Yihang Sun, Yixiao Wang, Jianglan Wei, Ziyang Gong, Xiangyu Zhao, Masayoshi Tomizuka, Xue Yang, Junchi Yan, et al. Interleave-vla: Enhancing robot manipulation with interleaved image-text instructions. In *CoRL*, 2025. 2, 3

[20] Hao Li, Shuai Yang, Yilun Chen, Xinyi Chen, Xiaoda Yang, Yang Tian, Hanqing Wang, Tai Wang, Dahua Lin, Feng Zhao, et al. Towards efficient and robust manipulation via multi-frame vision-language-action modeling. In *AAAI*, volume 40, pages 18388–18396, 2026. 2, 3, 4, 6, 7, 22

[21] 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. *arXiv preprint arXiv:2510.00695*, 2025. 2, 3

[22] Qingda Hu, Ziheng Qiu, Zijun Xu, Kaizhao Zhang, Xizhou Bu, Zuolei Sun, Bo Zhang, Jieru Zhao, Zhongxue Gan, and Wenchao Ding. Resolving state ambiguity in robot manipulation via adaptive working memory recoding. *IEEE Robotics and Automation Letters*, 2026. 2, 3

[23] Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. Memoryvla: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. In *ICLR*, 2026. 2, 3, 4, 6, 7, 8, 9, 15, 16, 18, 22

[24] Ajay Sridhar, Jennifer Pan, Satvik Sharma, and Chelsea Finn. Memer: Scaling up memory for robot control via experience retrieval. *arXiv preprint arXiv:2510.20328*, 2025. 2, 3, 4

[25] Zaijing Li, Bing Hu, Rui Shao, Gongwei Chen, Dongmei Jiang, Pengwei Xie, Jianye Hao, and Liqiang Nie. Global prior meets local consistency: Dual-memory augmented vision-language-action model for efficient robotic manipulation. In *CVPR*, pages 35135–35145, 2026. 2, 3

[26] Qixun Wang, Yang Shi, Yifei Wang, Yuanxing Zhang, Pengfei Wan, Kun Gai, Xianghua Ying, and Yisen Wang. Monet: Reasoning in latent visual space beyond image and language. In *CVPR*, pages 12030–12040, 2026. 2

[27] Zeyuan Yang, Xueyang Yu, Delin Chen, Maohao Shen, and Chuang Gan. Machine mental imagery: Empower multimodal reasoning with latent visual tokens. In *CVPR*, pages 33510–33520, 2026. 2

[28] Bangzheng Li, Ximeng Sun, Jiang Liu, Ze Wang, Jialian Wu, Xiaodong Yu, Hao Chen, Emad Barsoum, Muhao Chen, and Zicheng Liu. Latent visual reasoning. In *ICLR*, 2026. 2

[29] Shuanghao Bai, Jing Lyu, Wanqi Zhou, Zhe Li, Dakai Wang, Lei Xing, Xiaoguang Zhao, Pengwei Wang, Zhongyuan Wang, Cheng Chi, et al. Latent reasoning vla: Latent thinking and prediction for vision-language-action models. *arXiv preprint arXiv:2602.01166*, 2026. 2

[30] Guibin Zhang, Muxin Fu, and Shuicheng Yan. Memgen: Weaving generative latent memory for self-evolving agents. *arXiv preprint arXiv:2509.24704*, 2025. 2

[31] Xinlei Yu, Chengming Xu, Guibin Zhang, Zhangquan Chen, Yudong Zhang, Yongbo He, Peng-Tao Jiang, Jiangning Zhang, Xiaobin Hu, and Shuicheng Yan. Vismem: Latent vision memory unlocks potential of vision-language models. In *CVPR*, pages 31544–31555, 2026. 2

[32] Xinlei Yu, Zhangquan Chen, Yongbo He, Tianyu Fu, Guanting Dong, Cheng Yang, Chengming Xu, Yue Ma, Xiaobin Hu, Zhe Cao, et al. The latent space: Foundation, evolution, mechanism, ability, and outlook. *arXiv preprint arXiv:2604.02029*, 2026. 2

[33] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *CoRL*, pages 2165–2183, 2023. 2, 3

[34] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, pages 4195–4205, 2023. 2, 4

[35] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *NeurIPS*, volume 36, pages 44776–44791, 2023. 2, 7, 8, 9, 19, 21, 22

[36] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. In *CoRL*, pages 3705–3728. PMLR, 2025. 2, 6, 7, 8, 9, 21, 22

[37] Egor Cherepanov, Nikita Kachaev, Alexey Kovalev, and Aleksandr Panov. Memory, benchmark & robots: A benchmark for solving complex tasks with reinforcement learning. In *ICLR*, volume 2026, pages 9759–9815, 2026. 2, 7, 8, 19, 22

[38] Jiafei Duan, Wentao Yuan, Wilbert Pumacay, Yi Ru Wang, Kiana Ehsani, Dieter Fox, and Ranjay Krishna. Manipulate-anything: Automating real-world robots using vision-language models. In *CoRL*, pages 5326–5350, 2025. 3

[39] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. In *CoRL*, pages 4573–4602, 2025. 3

[40] Jiaming Liu, Hao Chen, Pengju An, Zhuoyang Liu, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, et al. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025. 3

[41] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *CVPR*, pages 1702–1713, 2025. 3, 7

[42] Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, et al. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025. 3

[43] Yuqi Wang, Xinghang Li, Wenxuan Wang, Junbo Zhang, Yingyan Li, Yuntao Chen, Xinlong Wang, and Zhaoxiang Zhang. Unified vision-language-action model. *arXiv preprint arXiv:2506.19850*, 2025. 3

[44] Juyi Lin, Amir Taherin, Arash Akbari, Arman Akbari, Lei Lu, Guangyu Chen, Taskin Padir, Xiaomeng Yang, Weiwei Chen, Yiqian Li, et al. Vote: vision-language-action optimization with trajectory ensemble voting. *arXiv preprint arXiv:2507.05116*, 2025. 3

[45] Songsheng Wang, Rucheng Yu, Zhihang Yuan, Chao Yu, Feng Gao, Yu Wang, and Derek F Wong. Spec-vla: speculative decoding for vision-language-action models with relaxed acceptance. In *EMNLP*, pages 26916–26928, 2025. 3

[46] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025. 3, 7

[47] Rongyu Zhang, Menghang Dong, Yuan Zhang, Liang Heng, Xiaowei Chi, Gaole Dai, Li Du, Dan Wang, Yuan Du, and Shanghang Zhang. Mole-vla: Dynamic layer-skipping vision language action model via mixture-of-layers for efficient robot manipulation. In *AAAI*, volume 40, pages 18764–18772, 2026. 3

[48] Hongyu Wang, Chuyan Xiong, Ruiping Wang, and Xilin Chen. Bitvla: 1-bit vision-language-action models for robotics manipulation. *arXiv preprint arXiv:2506.07530*, 2025. 3

[49] Junjie Wen, Yichen Zhu, Jinming Li, Zhibin Tang, Chaomin Shen, and Feifei Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control. In *CoRL*, pages 3094–3114, 2025. 3

[50] Yating Wang, Haoyi Zhu, Mingyu Liu, Jiange Yang, Hao-Shu Fang, and Tong He. Vq-vla: Improving vision-language-action models via scaling vector-quantized action tokenizers. In *ICCV*, pages 11089–11099, 2025. 3

[51] Shengliang Deng, Mi Yan, Songlin Wei, Haixin Ma, Yuxin Yang, Jiayi Chen, Zhiqi Zhang, Taoyu Yang, Xuheng Zhang, Wenhao Zhang, et al. Graspvla: a grasping foundation model pre-trained on billion-scale synthetic action data. *arXiv preprint arXiv:2505.03233*, 2025. 3

[52] 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. *arXiv preprint arXiv:2603.12942*, 2026. 3

[53] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. In *ICLR*, volume 2024, pages 26703–26721, 2024. 3

[54] Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In *ICLR*, volume 2025, pages 54277–54296, 2025. 3, 6, 22

[55] Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025. 3

[56] Max Sobol Mark, Jacky Liang, Maria Attarian, Chuyuan Fu, Debidatta Dwibedi, Dhruv Shah, and Aviral Kumar. Bpp: Long-context robot imitation learning by focusing on key history frames. *arXiv preprint arXiv:2602.15010*, 2026. 3

[57] Minghui Lin, Pengxiang Ding, Shu Wang, Zifeng Zhuang, Yang Liu, Xinyang Tong, Wenxuan Song, Shangke Lyu, Siteng Huang, and Donglin Wang. Hif-vla: Hindsight, insight and foresight through motion representation for vision-language-action models. In *CVPR*, pages 20732–20742, 2026. 4

[58] Huiwon Jang, Sihyun Yu, Heeseung Kwon, Hojin Jeon, Younggyo Seo, and Jinwoo Shin. Contextvla: Vision-language-action model with amortized multi-frame context. *arXiv preprint arXiv:2510.04246*, 2025. 4

[59] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *ICML*, 2024. 5

[60] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. 5

[61] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, pages 11975–11986, 2023. 5

[62] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 5

[63] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, pages 7132–7141, 2018. 5

[64] Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025. 6, 7

[65] Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, He Wang, Zhizheng Zhang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. In *NeurIPS*, pages 24195–24228, 2025. 6

[66] Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Frank Wang, and Fu-En Yang. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. In *NeurIPS*, pages 82782–82802, 2025. 6, 7

[67] Fei Ni, Zhuo Chen, Yifu Yuan, Zibin Dong, Xianze Yao, Shan Luo, Jianye Hao, Jiankang Deng, and Stefanos Zafeiriou. Semanticvla: Towards semantic reasoning over action memorization via synergistic explicit trace and latent action planning. In *CVPR*, pages 12237–12247, 2026. 6, 7

[68] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 7

[69] Jinliang Zheng, Jianxiong Li, Dongxiu Liu, Yinan Zheng, Zhihao Wang, Zhonghong Ou, Yu Liu, Jingjing Liu, Ya-Qin Zhang, and Xianyuan Zhan. Universal actions for enhanced embodied foundation models. In *CVPR*, pages 22508–22519, 2025. 7

[70] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025. 7, 19

[71] Chi-Pin Huang, Yunze Man, Zhiding Yu, Min-Hung Chen, Jan Kautz, Yu-Chiang Frank Wang, and Fu-En Yang. Fast-thinkact: Efficient vision-language-action reasoning via verbalizable latent planning. *arXiv e-prints*, pages arXiv–2601, 2026. 7

[72] Mengya Liu, Baoxiong Jia, Jiangyong Huang, Jingze Zhang, and Siyuan Huang. Lara: Latent action representation alignment for vision-language-action models. *arXiv preprint arXiv:2606.07100*, 2026. 7

[73] Chenghao Liu, Jiachen Zhang, Chengxuan Li, Zhimu Zhou, Shixin Wu, Songfang Huang, and Huiling Duan. Ttf-vla: Temporal token fusion via pixel-attention integration for vision-language-action models. In *AAAI*, volume 40, pages 18452–18459, 2026. 22

[74] Jiahui Zhang, Yurui Chen, Yueming Xu, Ze Huang, Yanpeng Zhou, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, Xingyue Quan, Hang Xu, et al. 4d-vla: Spatiotemporal vision-language-action pretraining with cross-scene calibration. In *NeurIPS*, volume 38, pages 33914–33937, 2025. 22

SUMMARY OF THE APPENDIX

This supplementary material provides additional details and analyses for the ICLR 2027 submission, titled *"Dual Latent Memory in Vision-Language-Action Models for Robotic Manipulation"*. The appendix is organized as follows:

- §A provides additional method details.
- §B summarizes the architectural and training hyperparameters.
- §C provides the evaluation protocols for real-world and simulated environments.
- §D reports additional quantitative results and ablation studies.
- §E provides the case study of the latent memory mechanism.
- §F compares `LaMem-VLA` with representative temporal-context VLA baselines.
- §G presents additional qualitative results in real-world and simulated environments.
- §H discusses the limitations and future work.

A ADDITIONAL METHOD DETAILS

A.1 DUAL MEMORY VAULT UPDATING

The construction of the two memory vaults is described in the main paper. Here, we provide the complete online updating procedure. Both vaults are initialized as empty and reset at the beginning of each episode. At each timestep t, `LaMem-VLA` first retrieves from the existing vaults and writes the current memory units only after predicting the corresponding action chunk. The short-term unit is the key–value pair $\boldsymbol{m}_{\mathrm{s}}^{t} = (\boldsymbol{k}_{\mathrm{s}}^{t}, \boldsymbol{v}_{\mathrm{s}}^{t})$ defined in Eq. 2, while the long-term unit is the current memory-grounded action representation $\boldsymbol{m}_{\mathrm{l}}^{t} = \boldsymbol{H}^{\text{action}}$ produced by the action-reasoning pass. Memory units are stored in chronological order, such that temporal adjacency is determined directly by their positions in the vault. The current units are first appended to their corresponding vaults:

$$\begin{aligned} \overline{\mathcal{M}}_{t}^{\text{short}} &= \mathrm{Append}\left(\mathcal{M}_{t-1}^{\text{short}}, \boldsymbol{m}_{\mathrm{s}}^{t}\right), \\ \overline{\mathcal{M}}_{t}^{\text{long}} &= \mathrm{Append}\left(\mathcal{M}_{t-1}^{\text{long}}, \boldsymbol{m}_{\mathrm{l}}^{t}\right). \end{aligned} \tag{10}$$

Let $n_{\mathrm{s}} = |\overline{\mathcal{M}}_{t}^{\text{short}}|$ and $n_{\mathrm{l}} = |\overline{\mathcal{M}}_{t}^{\text{long}}|$. Each vault has a maximum capacity of L and is consolidated independently only when its number of stored units exceeds this capacity.

Following the adjacent consolidation strategy of MemoryVLA [23], we identify the most similar pair of neighboring units in each overflowing vault. For the short-term vault, similarity is computed between the stored retrieval keys. For the long-term vault, each action representation is mean-pooled to obtain its comparison representation:

$$\begin{aligned} i_{\mathrm{s}}^{*} &= \underset{1 \le i < n_{\mathrm{s}}}{\arg\max} \cos\left(\boldsymbol{k}_{\mathrm{s}}^{i}, \boldsymbol{k}_{\mathrm{s}}^{i+1}\right), \\ i_{\mathrm{l}}^{*} &= \underset{1 \le i < n_{\mathrm{l}}}{\arg\max} \cos\left(\mathrm{MeanPool}(\boldsymbol{m}_{\mathrm{l}}^{i}), \mathrm{MeanPool}(\boldsymbol{m}_{\mathrm{l}}^{i+1})\right). \end{aligned} \tag{11}$$

Restricting consolidation to adjacent units preserves the chronological organization of each vault while removing locally redundant memory.

The selected units are merged using an unweighted arithmetic mean. For the short-term vault, the key and value are averaged separately, whereas the long-term action representations are averaged directly:

$$\begin{aligned} \widetilde{\boldsymbol{k}}_{\mathrm{s}} &= \frac{1}{2}\left(\boldsymbol{k}_{\mathrm{s}}^{i_{\mathrm{s}}^{*}} + \boldsymbol{k}_{\mathrm{s}}^{i_{\mathrm{s}}^{*}+1}\right), \\ \widetilde{\boldsymbol{v}}_{\mathrm{s}} &= \frac{1}{2}\left(\boldsymbol{v}_{\mathrm{s}}^{i_{\mathrm{s}}^{*}} + \boldsymbol{v}_{\mathrm{s}}^{i_{\mathrm{s}}^{*}+1}\right), \qquad \widetilde{\boldsymbol{m}}_{\mathrm{s}} = \left(\widetilde{\boldsymbol{k}}_{\mathrm{s}}, \widetilde{\boldsymbol{v}}_{\mathrm{s}}\right), \\ \widetilde{\boldsymbol{m}}_{\mathrm{l}} &= \frac{1}{2}\left(\boldsymbol{m}_{\mathrm{l}}^{i_{\mathrm{l}}^{*}} + \boldsymbol{m}_{\mathrm{l}}^{i_{\mathrm{l}}^{*}+1}\right). \end{aligned} \tag{12}$$

The retrieval representation of $\widetilde{\boldsymbol{m}}_{\mathrm{l}}$ is subsequently computed on demand using $\mathrm{MeanPool}(\widetilde{\boldsymbol{m}}_{\mathrm{l}})$, consistent with the long-term retrieval procedure in the main paper.

Let $\mathrm{Replace}(\mathcal{M}, i, \boldsymbol{m})$ denote replacing the i-th and $(i+1)$-th units in $\mathcal{M}$ with $\boldsymbol{m}$ while leaving all remaining units in their original order. The complete update is

$$\mathcal{M}_t^{\text{short}} = \begin{cases} \overline{\mathcal{M}}_t^{\text{short}}, & n_{\mathrm{s}} \le L, \\ \mathrm{Replace}\left(\overline{\mathcal{M}}_t^{\text{short}}, i_{\mathrm{s}}^*, \widetilde{\boldsymbol{m}}_{\mathrm{s}}\right), & n_{\mathrm{s}} = L+1, \end{cases}$$
$$\mathcal{M}_t^{\text{long}} = \begin{cases} \overline{\mathcal{M}}_t^{\text{long}}, & n_{\mathrm{l}} \le L, \\ \mathrm{Replace}\left(\overline{\mathcal{M}}_t^{\text{long}}, i_{\mathrm{l}}^*, \widetilde{\boldsymbol{m}}_{\mathrm{l}}\right), & n_{\mathrm{l}} = L+1. \end{cases} \tag{13}$$

Since exactly one unit is appended to each vault at every policy step, its size can exceed L by at most one. A single consolidation therefore restores the prescribed capacity.

A.2 Latent Memory Seeker and Condenser

Timestep-aware memory retrieval. Following MemoryVLA [23], each memory unit retains its episode timestep t_i. Before retrieval, we add a fixed sinusoidal timestep embedding $\mathrm{TE}(t_i)$ to the corresponding short- and long-term retrieval keys:

$$\bar{\boldsymbol{k}}_{\mathrm{s}}^i = \boldsymbol{k}_{\mathrm{s}}^i + \mathrm{TE}(t_i), \qquad \bar{\boldsymbol{k}}_{\mathrm{l}}^i = \mathrm{MeanPool}(\boldsymbol{m}_{\mathrm{l}}^i) + \mathrm{TE}(t_i). \tag{14}$$

The augmented keys replace the original retrieval keys in the similarity scores, while the stored value tokens remain unchanged. This encoding allows the seeker to distinguish similar memory content occurring at different stages of an episode.

Query Builder $\mathcal{B}$. We implement the query builder $\mathcal{B}$ as a lightweight two-layer Transformer that converts the current multimodal cognition representation into a fixed-length retrieval query. Given $\boldsymbol{Q}_t^{\text{con}} \in \mathbb{R}^{N_{\text{con}} \times C}$, we append the learnable query slots $\boldsymbol{Q}^{\text{init}} \in \mathbb{R}^{K_{\mathrm{q}} \times C}$ to form the input sequence:

$$\boldsymbol{U}_{\mathcal{B}}^0 = [\boldsymbol{Q}_t^{\text{con}}; \boldsymbol{Q}^{\text{init}}], \qquad \boldsymbol{Q}_t = \mathcal{B}\left(\boldsymbol{U}_{\mathcal{B}}^0; \boldsymbol{A}_{\mathcal{B}}\right)[-K_{\mathrm{q}}:]. \tag{15}$$

Each layer follows a standard pre-normalized Transformer block, consisting of masked multi-head self-attention and a position-wise feed-forward network with residual connections. The attention and feed-forward parameters are specific to each module and Transformer layer.

The attention mask $\boldsymbol{A}_{\mathcal{B}}$ allows the appended query slots to read the complete multimodal context while preventing the original context tokens from attending to these slots. With rows indexing attention queries and columns indexing attention keys, the mask is

$$(\boldsymbol{A}_{\mathcal{B}})_{ij} = \begin{cases} -\infty, & 1 \le i \le N_{\text{con}}, \quad N_{\text{con}} < j \le N_{\text{con}} + K_{\mathrm{q}}, \\ 0, & \text{otherwise}. \end{cases} \tag{16}$$

Thus, $\boldsymbol{Q}^{\text{init}}$ can attend to $\boldsymbol{Q}_t^{\text{con}}$ and to the other query slots, whereas $\boldsymbol{Q}_t^{\text{con}}$ can attend only within the original context sequence. In implementation, $-\infty$ is replaced by a sufficiently large negative value. After the final layer, the outputs corresponding to the last K_{q} positions are retained as $\boldsymbol{Q}_t$.

Memory Formers $\mathcal{F}_{\mathbf{v}}$ and $\mathcal{F}_{\mathbf{c}}$. The visual memory former $\mathcal{F}_{\mathrm{v}}$ and semantic memory former $\mathcal{F}_{\mathrm{c}}$ follow the same two-layer Transformer architecture as $\mathcal{B}$. The two formers do not share parameters, and $\mathcal{B}$ is instantiated separately. Moreover, $\mathcal{F}_{\mathrm{v}}$ and $\mathcal{F}_{\mathrm{c}}$ use distinct learnable target sequences $\boldsymbol{T}_{\mathrm{s}} \in \mathbb{R}^{L_{\mathrm{s}} \times C}$ and $\boldsymbol{T}_{\mathrm{l}} \in \mathbb{R}^{L_{\mathrm{l}} \times C}$, respectively. Their inputs and outputs are

$$\begin{aligned} \boldsymbol{U}_{\mathrm{v}}^0 &= [\boldsymbol{Q}_t; \boldsymbol{Z}^{\text{short}}; \boldsymbol{T}_{\mathrm{s}}], \quad \boldsymbol{M}^{\text{short}} = \mathcal{F}_{\mathrm{v}}\left(\boldsymbol{U}_{\mathrm{v}}^0; \boldsymbol{A}_{\mathrm{v}}\right)[-L_{\mathrm{s}}:], \\ \boldsymbol{U}_{\mathrm{c}}^0 &= [\boldsymbol{Q}_t; \boldsymbol{Z}^{\text{long}}; \boldsymbol{T}_{\mathrm{l}}], \quad \boldsymbol{M}^{\text{long}} = \mathcal{F}_{\mathrm{c}}\left(\boldsymbol{U}_{\mathrm{c}}^0; \boldsymbol{A}_{\mathrm{c}}\right)[-L_{\mathrm{l}}:]. \end{aligned} \tag{17}$$

Let $N_{\mathrm{v}} = |[\boldsymbol{Q}_t; \boldsymbol{Z}^{\text{short}}]|$ and $N_{\mathrm{c}} = |[\boldsymbol{Q}_t; \boldsymbol{Z}^{\text{long}}]|$ denote the numbers of content tokens supplied to the two formers, and let $L_{\mathrm{v}} = L_{\mathrm{s}}$ and $L_{\mathrm{c}} = L_{\mathrm{l}}$. For $x \in \{\mathrm{v}, \mathrm{c}\}$, the corresponding attention mask is

$$(\boldsymbol{A}_x)_{ij} = \begin{cases} -\infty, & 1 \le i \le N_x, \quad N_x < j \le N_x + L_x, \\ 0, & \text{otherwise}. \end{cases} \tag{18}$$

Table 6: Architecture and memory hyper-parameters of **LaMem-VLA** (§B.1).

Hyper-parameter	Value
Backbone and input	
Vision-language backbone	Prismatic VLM
VLM parameter count	7B
Visual encoders	DINOv2 and SigLIP
VLA hidden dimension C	4096
Camera view	Single third-person RGB
Model input resolution	224×224
Raw real-world image resolution	640×480
Visual-token length N_{i}	128
Action dimension	7-DoF
Action horizon H	16
Number of action queries N_{a}	4
Diffusion action expert	DiT-L
Diffusion expert parameter count	Approximately 300M
Memory curator and vaults	
Short-term compressor $\mathcal{C}_{\mathrm{s}}$	SE-bottleneck
SE reduction ratio	16
Capacity per vault L	16
Memory write frequency	One unit per policy step
Memory seeker and condenser	
Retrieval budget K	8
Number of query slots K_{q}	8
Short-term output length L_{s}	8 tokens
Long-term output length L_{l}	4 tokens
Depth of $\mathcal{B}$	2 Transformer layers
Depth of $\mathcal{F}_{\mathrm{v}}$ and $\mathcal{F}_{\mathrm{c}}$	2 Transformer layers each

Under this mask, the context query $\boldsymbol{Q}_t$ and the retrieved memory tokens can attend to one another. The target slots can attend to $\boldsymbol{Q}_t$, all retrieved memory tokens, and the other target slots, while neither $\boldsymbol{Q}_t$ nor the retrieved memory tokens can attend back to the target slots. The outputs at the target positions therefore provide fixed-length latent memory tokens conditioned jointly on the current multimodal context and the retrieved historical evidence.

B Implementation and Training Details

B.1 Hyper-parameters

We summarize the principal architectural and optimization hyper-parameters of **LaMem-VLA** in Tables 6 and 7. Unless otherwise specified, the same model and memory configurations are used across all benchmarks. We instantiate **LaMem-VLA** with a 7B-parameter Prismatic VLM and a diffusion action expert containing approximately 300M parameters. The policy receives a single third-person RGB observation resized to 224×224 and predicts a chunk of 16 continuous 7-DoF actions. Both memory vaults have a capacity of $L = 16$, from which the memory seeker retrieves $K = 8$ units per vault. The memory condenser produces $L_{\mathrm{s}} = 8$ short-term and $L_{\mathrm{l}} = 4$ long-term latent memory tokens. The query builder $\mathcal{B}$ and memory formers $\mathcal{F}_{\mathrm{v}}$ and $\mathcal{F}_{\mathrm{c}}$ each contain two Transformer layers and are parameterized independently. During inference, the diffusion action expert uses DDIM sampling with 10 denoising steps.

B.2 Training Setup

Offline demonstrations are organized into episode-consistent sequences rather than independent frames. Within each sequence, frames are processed in temporal order while the two vault states are

Table 7: Optimization and diffusion hyper-parameters of **LaMem-VLA** (§B.1).

Hyper-parameter	Value
Optimization	
Training accelerators	8 NVIDIA H800 GPUs
Distributed training strategy	PyTorch FSDP
Batch size per GPU	32
Global batch size	256
Optimizer	AdamW
Learning rate	2×10^{-5}
Maximum gradient norm	1.0
Exponential moving average	Disabled
Diffusion action generation	
Training objective	Mean squared error
Repeated diffusion samples per training input	4
Training diffusion timesteps	100
Noise schedule	Squared cosine
Inference sampler	DDIM
Inference denoising steps	10
Classifier-free guidance scale	1.5
DDIM stochasticity η	0.0
Denoised-output clipping	Disabled

carried forward; the vaults are reset at episode or independently sampled group boundaries. Thus, each ordered sequence and its evolving vault states form one stateful memory training sample.

SimplerEnv-Bridge. We train on Bridge v2 for 50K steps using a stream dataloader, following the episode-aware design of MemoryVLA [23]. Each demonstration is unpacked into consecutive frames tagged with its episode ID. A minibatch is filled sequentially from the same episode whenever possible. If an episode ends before the minibatch is full, the remaining positions are filled from the next episode and the vaults are reset at the episode boundary. The next minibatch resumes from the previous stream cursor, preserving the temporal order within each episode. The memory length is set to 16.

LIBERO. Following OpenVLA [2], we use 50 demonstrations per task. Spatial, Object, and Goal are trained separately for 20K steps each, whereas Long-10 and Long-90 are trained jointly for 40K steps. Each minibatch is divided into temporal groups, with each group containing 16 frames sampled from a single episode and sorted by their original timestamps. The vault states are propagated within each group and reset between groups. The memory length is set to 16.

MIKASA-Robo. We jointly train the five tasks on all 1,250 demonstrations for 20K steps, using 128×128 RGB observations and delta end-effector control. We adopt the same grouped dataloader as LIBERO and set the memory length to 16.

Real-world. Models are trained for 5K–20K steps depending on the task. The General Tasks and Long-horizon Tasks contain 200 and 300 demonstrations per task, respectively. The memory length is set to 16 for general tasks and 64 for long-horizon temporal tasks.

B.3 Data Augmentation

During training, each third-person RGB observation is processed by a fixed sequence of stochastic augmentations: random resized cropping, followed by brightness, contrast, saturation, and hue jittering. The crop covers 90% of the original image area with a fixed aspect ratio of 1.0 and is subsequently resized to 224×224. We set the brightness jitter magnitude to 0.2, sample the contrast and saturation factors uniformly from $[0.8, 1.2]$, and sample the hue offset uniformly from $[-0.05, 0.05]$. All augmentations are disabled during evaluation.

Table 8: Overview of *General Tasks* on Real-World (§C.4).

	General Tasks		
Task name	Stack cup	Drawer	Pick fruit
Prompt	Stack one cup on top of another cup.	Open the drawer, put the items on the table inside, and close the drawer.	Place the banana, kiwi, and wax apple into the basket in that order.
# demonstrations	200	200	200
# rollouts	50	50	50
Performance (%)	82	78	72

Table 9: Overview of *Long-horizon Tasks* on Real-World (§C.4).

	Long-horizon Tasks		
Task name	Cover block	Press buttons	Fill drawer
Prompt	Cover the building block with a cup, then lift up the cup covering the block.	Press and hold the red and blue buttons in sequence for a while.	Put the three items on the table into the drawer, then close the drawer.
# demonstrations	300	300	300
# rollouts	50	50	50
Performance (%)	56	66	62

C Additional Evaluation Details

C.1 Evaluation on SimplerEnv-Bridge

We follow the official CogACT evaluation protocol [15], including its evaluation scripts and adaptive action ensemble strategy, with $\alpha = 0.1$ and an ensemble horizon of 7. We train each model for 50K steps and evaluate every 2.5K steps, selecting the checkpoint with the best validation performance since the diffusion training objective is not a reliable proxy for policy performance. The selected checkpoint is evaluated over 24 trials per task. We additionally re-evaluate the released CogACT-Large checkpoint under the same protocol, as its per-task results were not provided in the original paper. For π_0, we report the float32 results from the open-source reproduction with both uniform and Beta timestep sampling.

C.2 Evaluation on MIKASA-Robo

For MIKASA-Robo [37], we evaluate each task over 100 episodes. Validation is performed every 1K training steps, and we report the success rate achieved by the checkpoint with the best validation performance. As in LIBERO, we do not use action ensembling in our MIKASA-Robo experiments.

C.3 Evaluation on LIBERO

We evaluate on all five LIBERO suites [35], including Spatial, Object, Goal, Long-10, and Long-90. Models are validated every 1K steps, and each task is evaluated over 50 trials, with success rates reported at the best validation checkpoint. Unlike SimplerEnv-Bridge, we do not apply action ensembling on LIBERO. For a fair comparison, we reproduce CogACT [15] using its official codebase, while results for π_0 [1] and π_0-FAST [70] are taken from their reported numbers. Notably, both π_0 variants additionally use wrist-camera observations and proprioceptive states, whereas our method relies solely on a single third-person RGB stream. Despite this more restricted observation setting, our method achieves higher success rates across the evaluated suites.

Table 10: Task descriptions (§C.4).

Task	Description
Stack cup	Pick up one paper cup and stack it on top of the other cup.
Drawer	Open the drawer, put the items on the table into it, and close the drawer.
Pick fruit	Pick up the banana, the kiwi, and the wax apple and place them into the basket in this order.
Cover block	Pick up a cup, place it upside down over the building block so that the block is hidden, and then lift the same cup to reveal the block.
Press buttons	Press and hold the red button for a while, and then press and hold the blue button for a while.
Fill drawer	Put the three items on the table into the open drawer one by one, and then close the drawer.

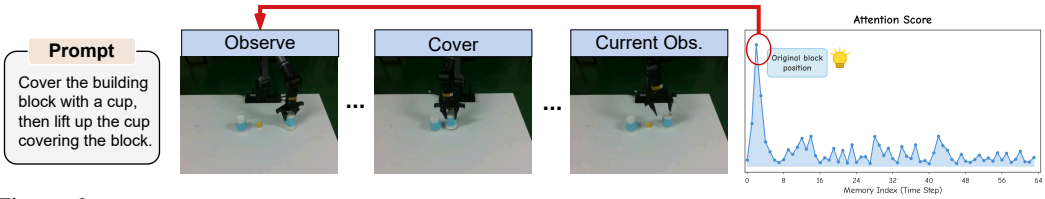


Figure 6: **Visualization of the memory mechanism in the real-world Cover Block task** (§E). `LaMem-VLA` attends to historical frames revealing the original block location, providing temporal context for lifting the cup covering the now-occluded block.

C.4 Evaluation on Real-world

Platform. We use the AgileX Piper, a 6-DoF desktop manipulator mounted on a fixed table. Perception is provided by one exocentric RealSense D435i RGB camera placed in front of the workspace; depth is not used. Demonstrations are collected by human teleoperation at 30 Hz and stored as raw 640×480 frames with synchronized joint positions, end-effector poses, and gripper openings. During deployment, the policy receives the current RGB frame and the language instruction, and predicts an action chunk of 7-D end-effector deltas (translation, rotation, gripper), which are converted to joint targets by inverse kinematics and executed at 30 Hz.

Task Suites. As summarized in Tables 8 and 9, we construct two real-world task suites: General Tasks and Long-horizon Tasks. General Tasks evaluate standard pick-and-place and articulated-object manipulation under randomized object configurations, where task progress can be inferred directly from the current observation. In contrast, Long-horizon Tasks assess the ability to preserve and exploit historical context, as at least one decision cannot be resolved from the current observation alone due to object occlusion, visually aliased task states, or elapsed-time-dependent requirements. For each evaluation episode, object poses are randomly initialized within the workspace. The test-time language instruction is identical to that used during data collection. Detailed task descriptions and prompts are provided in Table 10, with additional qualitative results presented in the next section.

D Additional Ablation Study

Inference Efficiency. We further compare the inference efficiency of `LaMem-VLA` with its CogACT baseline [15] under the same inference configuration. As shown in Table 11, `LaMem-VLA` requires 0.315s to generate a 16-step action chunk, corresponding to an action-generation throughput of 50.8Hz. Compared with CogACT, the incorporation of latent memory increases latency by 0.076s and reduces throughput by 16.1Hz. Meanwhile, peak GPU memory increases from 15.9GB to 17.8GB, corresponding to an additional overhead of 1.9GB. The overall computational overhead remains controlled by the bounded design of our latent memory. Moreover, adjacent similarity merging prevents the vault size from growing with the trajectory length. These results show that `LaMem-VLA` introduces long-horizon memory reasoning into the VLA model while retaining an action throughput of 50.8Hz with controlled additional latency and GPU memory consumption.

Table 11: **Inference efficiency comparison on an NVIDIA H20 GPU (§D).** Latency and peak GPU memory are averaged over 300 runs in bfloat16 with a batch size of 1 and an action chunk size of 16. Throughput denotes the number of actions generated per second.

Model	Latency (s) ↓	Throughput (Hz) ↑	Peak Memory (GB) ↓
CogACT [15] (Baseline)	0.239s	66.9Hz	15.9GB
`LaMem-VLA` (Ours)	0.315s	50.8Hz	17.8GB

Memory Bank Size L. We next investigate the impact of memory bank size L, which controls the maximum number of units maintained in each memory vault. As shown in Fig. 7, increasing L from 4 to 16 consistently improves the success rate from 93.0% to 95.4% on Long-10 [35] and from 68.7% to 73.9% on SimplerEnv [36]. This supports our hypothesis that sufficient capacity is essential for preserving diverse historical evidence and task-progress cues, whereas an overly small memory bank compresses the history too aggressively. However, further increasing L to 32 or 64 yields only a marginal gain of 0.4 points on Long-10 and instead decreases performance by 2.0 points on SimplerEnv. We speculate that an excessively large memory bank delays the consolidation of locally redundant states and introduces unnecessary candidates into memory retrieval, without providing additional decision-relevant evidence. Therefore, we set $L = 16$ as the default configuration, which strikes a favorable balance between historical information preservation, retrieval effectiveness, and memory efficiency.

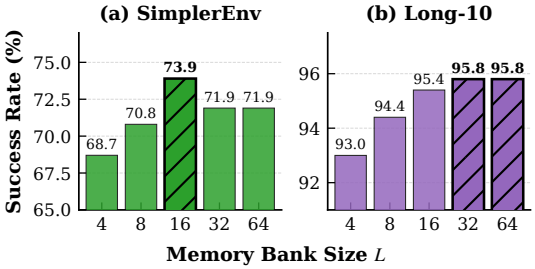


Figure 7: **Ablation study of memory bank size L** on SimplerEnv [36] and LIBERO-long [35] (§D).

Memory Vault Updating Strategy. We further investigate the impact of different memory vault updating strategies. We compare our strategy with two alternatives: FIFO discards the earliest memory unit once the vault reaches its capacity, while Random Merge randomly selects memory units for consolidation without considering their representation similarity. All variants use the same fixed capacity for both memory vaults. As shown in Fig. 8, our adjacent similarity merging consistently achieves the best performance across all five LIBERO suites [35]. Notably, larger gains are observed on Goal, Long-10, and Long-90, where `LaMem-VLA` outperforms FIFO by **4.4**, **5.2**, and **3.4** points and Random Merge by **4.0**, **5.4**, and **3.0** points, respectively. FIFO may prematurely remove early evidence that remains relevant to subsequent decisions, whereas Random Merge may combine distinct historical states and thereby obscure informative temporal cues. In contrast, our strategy consolidates the most similar pair among temporally adjacent memory units, preferentially removing local redundancy while retaining distinctive state transitions. These results verify that similarity-aware local consolidation more effectively preserves task-relevant historical information, particularly when action prediction depends on task progress over extended horizons.

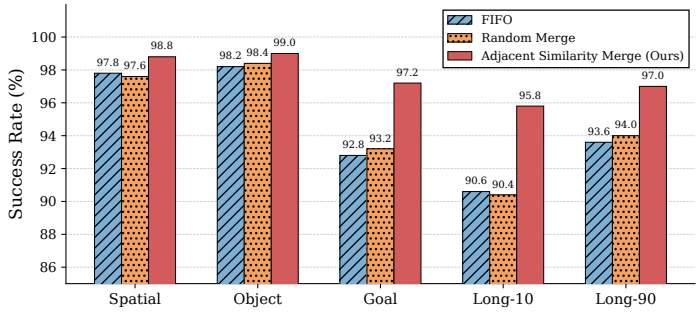


Figure 8: **Ablation study of different memory vault updating strategies** on the five LIBERO suites [35] (§D). All methods use the same fixed memory capacity.

E Case Study of Memory Mechanism

Fig. 6 visualizes the retrieved memory elements and their attention weights in the real-world Cover Block task. Once the block is covered by a cup, `LaMem-VLA` assigns high attention to earlier frames revealing its original location, preserving the correspondence between the hidden block and the

Table 12: **Quantitative comparison results with temporal-context VLA models across three simulation benchmarks [35, 36, 37] (§F).** Success rates (%) are reported. The LIBERO average is computed over the first four suites because most VLA models do not report results on Long-90.

(a) SimplerEnv-Bridge

Method	Publication	Spoon on Towel	Carrot on Plate	Stack Cube	Eggplant in Basket	Avg. Success
TraceVLA [54]	ICLR2025	12.5	16.6	16.6	65.0	27.7
RoboVLMs [18]	ArXiv2024	45.8	20.8	4.2	79.2	37.5
CronusVLA [20]	AAAI2026	66.7	54.2	20.8	**100.0**	60.4
MemoryVLA [23]	ICLR2026	75.0	**75.0**	37.5	**100.0**	71.9
LaMem-VLA (Ours)	–	**83.3**	**75.0**	**41.7**	95.8	**73.9**

(b) LIBERO

Method	Publication	Spatial	Object	Goal	Long-10	Avg. Success
TTF-VLA [73]	AAAI2026	84.0	73.0	81.0	58.0	74.0
TraceVLA [54]	ICLR2025	84.6	85.2	75.1	54.1	74.8
4D-VLA [74]	NeurIPS2025	88.9	95.2	90.9	79.1	88.6
MemoryVLA [23]	ICLR2026	98.4	98.4	96.4	93.4	96.7
CronusVLA [20]	AAAI2026	97.3	**99.6**	96.9	94.0	97.0
LaMem-VLA (Ours)	–	**98.8**	99.0	**97.2**	**95.8**	**97.7**

(c) MIKASA-Robo

Method	Publication	ShellGame Touch	Intercept Medium	Remb. Color-3	Remb. Color-5	Remb. Color-9	Avg. Success
CronusVLA [20]	AAAI2026	32	5	31	13	9	18.0
MemoryVLA [23]	ICLR2026	88	24	44	30	20	41.2
LaMem-VLA (Ours)	–	**94**	**43**	**58**	**37**	**25**	**51.4**

covering cup. This demonstrates that **LaMem-VLA** retrieves decision-relevant historical information to guide subsequent actions when critical visual information is absent from the current observation.

F Comparison with Temporal-Context VLA Baselines

We conduct a focused comparison with representative temporal-context VLAs that explicitly exploit historical observations or memory. As summarized in Table 12, **LaMem-VLA** consistently achieves the best average performance across SimplerEnv-Bridge [36], LIBERO [35], and Mikasa-Robo [37]. On SimplerEnv-Bridge, it reaches 73.9%, outperforming MemoryVLA [23] and CronusVLA [20] by **2.0** and **13.5** points, respectively. On LIBERO [35], **LaMem-VLA** achieves 97.7% over the four commonly reported suites, exceeding the strongest temporal-context baseline by **0.7** points. More notably, on the memory-intensive Mikasa-Robo benchmark [37], it improves over MemoryVLA and CronusVLA by **10.2** and **33.4** points, respectively. These consistent gains across diverse temporal dependencies substantiate the effectiveness of integrating historical evidence as context-native latent memory for action reasoning.

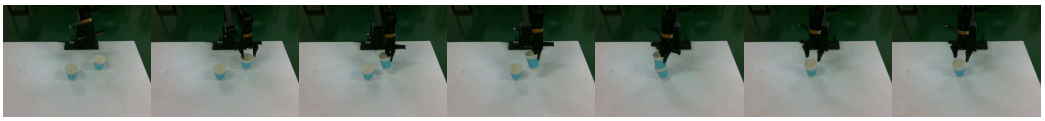
G Qualitative Results

G.1 Real-world Evaluation

We present qualitative real-world results of **LaMem-VLA**. By weaving complementary short-term visual and long-term semantic memories into VLA reasoning, **LaMem-VLA** exhibits superior performance on both General Tasks (Fig. 9) and Long-horizon Tasks (Fig. 10).

G.2 Simulation Evaluation

Qualitative results in simulated environments are presented in Figs. 11. Specifically, Fig. 11 provides representative trajectories on LIBERO [35], spanning all five suites.



(a) Stack cup: Pick up one paper cup and stack it on top of the other cup.

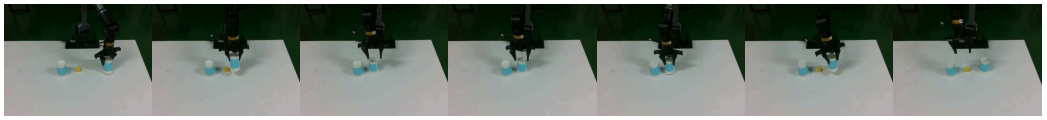


(b) Drawer: Open the drawer, put the items on the table into it, and close the drawer.

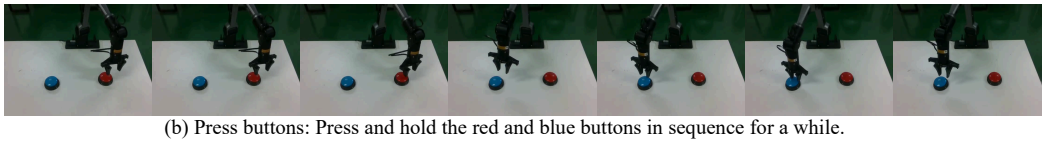


(c) Pick fruit: Place the banana, kiwi, and wax apple into the basket in that order.

Figure 9: **Qualitative results of `LaMem-VLA` on Real-World** (§G). From top to bottom, we illustrate three General Tasks: *Stack cup*, *Drawer*, and *Pick fruit*.



(a) Cover block: Cover the building block with a cup, then lift up the cup covering the block.



(b) Press buttons: Press and hold the red and blue buttons in sequence for a while.

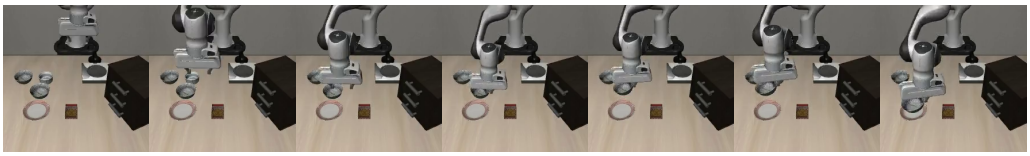


(c) Fill drawer: Put the three items on the table into the drawer, then close the drawer.

Figure 10: **Qualitative results of `LaMem-VLA` on long-horizon tasks** (§G). Representative examples include *Cover block*, *Press buttons*, and *Fill drawer*.

H Limitation and Future Work

Despite its effectiveness, **`LaMem-VLA`** has two limitations that merit further investigation. **First**, the current memory curator operates with fixed storage and retrieval budgets, and consolidates adjacent memory units according to their representation similarity. This design keeps the memory overhead bounded, but implicitly treats similarity as a proxy for future utility. In long or event-dense trajectories, evidence that appears redundant when stored may later become essential for resolving task progress, while irreversible consolidation makes such information difficult to recover. A promising direction is to develop task-conditioned memory management that jointly determines what to retain, merge, and retrieve according to the estimated decision value and uncertainty of each memory unit. Dynamically allocating memory capacity based on trajectory complexity may further improve the balance between information preservation and efficiency. **Second**, the current memory is confined to a single episode and reset once the episode terminates. This prevents information leakage across trajectories, but also means that the policy cannot reuse experience accumulated from previous interactions. Future work may extend **`LaMem-VLA`** with persistent experiential memory, yielding a hierarchical memory system that couples within-episode working memory with reusable cross-episode experience. Such an extension requires reliable mechanisms for online consolidation, selective forgetting, and uncertainty-aware retrieval, so that relevant experience can facilitate new tasks without introducing negative transfer from outdated or mismatched memories.



(a) Spatial: Pick up the black bowl between the plate and the ramekin and place it on the plate.



(b) Object: Pick up the alphabet soup and place it in the basket.



(c) Goal: Open the middle drawer of the cabinet.



(d) Long: Put both the alphabet soup and the tomato sauce in the basket.



(e) LIBERO-90: Close the top drawer of the cabinet and put the black bowl on top of it.

Figure 11: **Qualitative results of `LaMem-VLA` on LIBERO tasks** (§G). Representative examples include tasks from Spatial, Object, Goal, Long-10, and Long-90 suites.