# Native Memory Compression for Long-Horizon Robotic Manipulation

**Ziye Wang**[1] **Chaojun Ni**[2] **Modi Shi**[3] **Jiazhi Yang**[4] **Mengdi Li**[5] **Zhizhong Su**[5]
**Tianwei Lin**[5] **Hongyang Li**[1†]

[1] The University of Hong Kong [2] Peking University [3] Beihang University
[4] The Chinese University of Hong Kong [5] Horizon Robotics

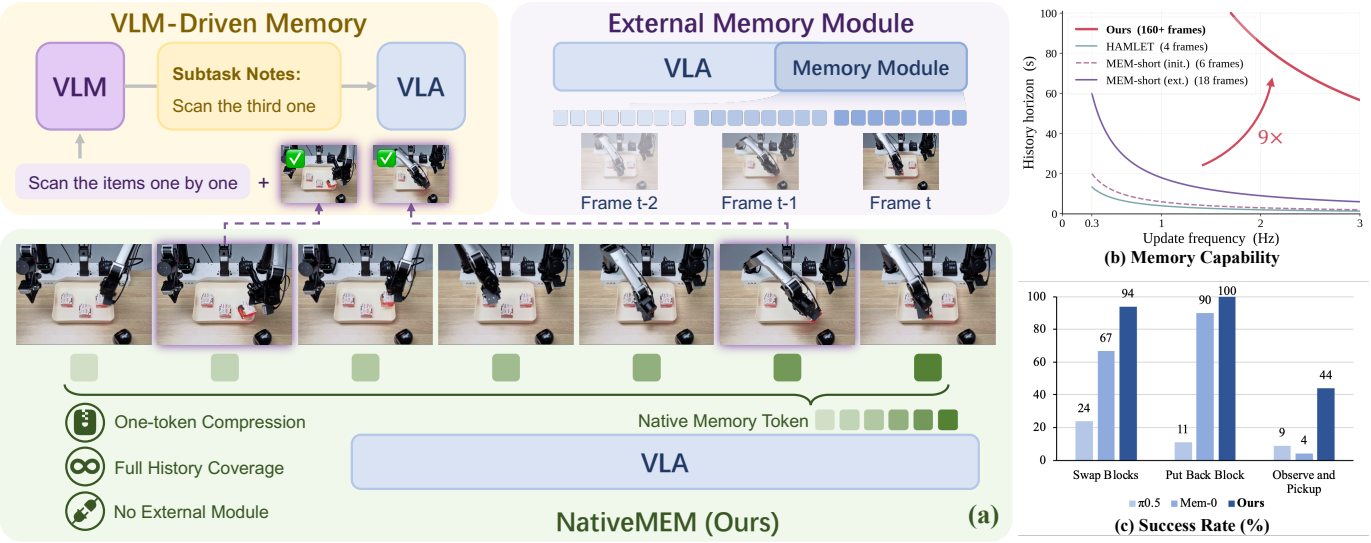



Fig. 1: **NativeMEM** differs from prior memory-augmented VLAs that rely on VLM-generated textual notes or external memory modules. (a) By repurposing the VLA's own vision encoder, it compresses each historical frame-view observation into a single native memory token, allowing the policy to condition on the full visual history through its original token sequence. (b) This ultra-compact representation enables minute-level histories with over 160 frames, providing a $9\times \sim 40\times$ longer history horizon than prior methods. (c) NativeMEM achieves the highest success rates across memory-dependent manipulation tasks.

***Abstract*—How can pretrained Vision-Language-Action (VLA) models retain long-horizon visual histories with high-frequency updates without sacrificing efficiency? Existing approaches rely on external memory management, which restrains either the memory horizon or the reactiveness of pretrained policies. To this end, we present NativeMEM, a VLA policy that features long-term and real-time updated memory. At its core is an efficient memory encoding scheme, Native Memory Compression, which repurposes the VLA's own vision encoder to compress each historical frame from each camera view into a single token. Appended to the input sequence, these memory tokens enable the pretrained VLA to attend over long-term history with negligible latency overhead, requiring neither an external planner nor a freshly initialized memory module. To align the memory tokens with the pretrained policy, we first develop a generic memory tokenizer under the supervision of a frozen VLA on memory-demanding data, and then unfreeze the VLA for task-specific fine-tuning. NativeMEM consistently outperforms prior methods, boosting success rates from 32.4% to 84.0% in simulation and up to 98.7% on real robots, while maintaining low inference latency and GPU memory usage. Notably, NativeMEM exhibits high data efficiency by achieving competitive results with prior arts using only 20% of the training data.**

†Corresponding author: Hongyang Li `hongyang@hku.hk`

## I. Introduction

Vision-Language-Action (VLA) models [53, 17, 3, 15, 6, 35, 31, 45, 16, 49] extend Vision-Language Models [2, 28, 1, 52] from perception to embodied decision-making, emerging as a promising paradigm for generalist robot control. Despite this progress, today's VLAs largely act without remembering. Most large-scale pretrained VLA policies remain reactive, predicting actions from only the current observation and language instruction. This single-frame formulation is insufficient for memory-dependent manipulation, where the correct action may depend on task progress, previously manipulated objects, operation counts, occluded states, or failed attempts. Formally, such tasks are partially observable or non-Markovian under a single-frame observation, calling for memory mechanisms that provide action-relevant historical visual context beyond the current scene.

To address this challenge, recent works have explored two main paradigms for memory-augmented VLAs. The first builds memory outside the policy, where a high-level VLM retrieves past keyframes and plans sparse subtasks for a low-

level VLA controller [48, 34, 38, 9, 40, 39, 46]. While effective for extending temporal context, such systems often turn memory into text format. However, such subtask descriptions require costly annotations for training, and subtle details are difficult to faithfully encode in language only. Another line of work builds memory with freshly initialized modules inside the policy, through recurrent states [20], compressed histories [19, 16, 26, 27, 29], or retrieval-augmented memory banks [11, 43, 12, 10, 25, 24]. Since these newly introduced modules are unseen during policy pretraining, this paradigm not only increases the overall architectural complexity, but also introduces the risk of performance degradation of the pretrained VLA. More fundamentally, they face a compression dilemma: without sufficient compression, fine-grained temporal histories are too costly to retain in full; with excessive or poorly structured compression, memory may discard details critical for action. This dilemma raises a central question: **how can a VLA retain long-horizon, fine-grained histories with minimal memory cost, while remaining compatible with the pretrained policy it builds upon?**

We address this question with **Native Memory Compression**, which compresses history directly into the pretrained VLA's own visual token space. Instead of introducing a separate memory architecture, NATIVEMEM repurposes the VLA's own vision encoder to distill past observations into compact memory tokens. We push compression to its token-efficient extreme: each historical frame is represented by a single native memory token, as shown in Fig. 1(a). Because these tokens are produced by the same visual pathway and injected into the original token sequence used by the pretrained policy, they can be interpreted as native visual evidence rather than external memory states, allowing the VLA to access long-horizon histories through its original attention mechanisms.

To learn such memory tokens so that they are both compact and aligned with the pretrained knowledge, we introduce a two-stage training pipeline that separates learning how to summarize memory from learning how to use it for a target task. In the first stage, we freeze the pretrained VLA and train a memory tokenizer derived from its native vision encoder. Given past observations, the tokenizer compresses each frame-view pair into a single visual summary token, which is appended to the original VLA token sequence. The tokenizer is optimized through the VLA's native action prediction loss, making the summaries action-supervised rather than reconstruction-supervised. By training on a mixture of standard manipulation data and memory-demanding tasks, this stage learns a general memory tokenizer whose outputs remain aligned with the pretrained VLA's token space. In the second stage, we freeze the learned tokenizer and perform task-specific full VLA finetuning with compact memory tokens, converting a pretrained single-frame VLA into a memory-augmented policy using limited task-specific demonstrations.

We extensively evaluate NATIVEMEM on manipulation tasks demanding long-horizon and adaptive memory understanding in both the simulation and the real world. Compared with pretrained memory-free VLA policies and prior memory-augmented baselines, NATIVEMEM achieves the highest task success rates while efficiently attending to minute-level histories with compact native memory, as shown in Fig. 1(b) and (c). Notably, even on unseen real-robot memory tasks, NATIVEMEM converts a pretrained single-frame VLA into a memory-augmented policy using only 100 task-specific demonstrations and approximately 5 hours of finetuning. Beyond this conversion efficiency, NATIVEMEM also exhibits strong data efficiency, achieving competitive performance with prior state-of-the-art methods while using only 20% of their training data.

The main contributions of this paper are summarized as follows: Our contributions are threefold:

- We propose NATIVEMEM, a memory-augmented VLA that uses Native Memory Compression to turn each historical frame from each camera view into a single visual memory token, enabling minute-level fine-grained visual histories.
- We introduce a two-stage training pipeline that first learns a memory tokenizer via the pretrained VLA's native action objective, aligning compact memory tokens with visual-action priors and enabling data-efficient task-specific finetuning from limited demonstrations.
- NATIVEMEM improves average success rates from 32.4% to **84.0%** in simulation and from 34.7% to **98.7%** on real robots, and matches leading memory-designed methods with only **20%** of the training data.

## II. RELATED WORK

**Vision-Language-Action Models.** Current Vision-Language-Action (VLA) models [3, 15, 35, 47, 42, 37, 36, 44, 21, 23, 41, 50] extend Vision-Language Models (VLMs) [2, 28, 1, 8] to embodied decision making by grounding language instructions in visual observations and predicting robot actions. RT-1 [5], RT-2 [53], OpenVLA [17], and $\pi_0$ [3] have advanced the field through large-scale robot data, vision-language pretraining, open-source generalist policies, and continuous diffusion-based action generation. However, most VLA models condition on the current observation and short-term history. Autoregressive VLAs [17, 18, 7, 14] can encode previous visual, language, or action tokens as implicit memory, but such memory remains constrained by context length and lacks an explicit mechanism for preserving task-relevant information over long-horizon interactions.

**VLM-Driven Memory with VLAs.** To enhance memory in VLA models, recent methods often combine a high-level VLM-based memory or planning module with a low-level VLA controller [34, 38, 9, 32]. Representative examples include MemER [34], which retrieves task-relevant keyframes to guide a VLA policy via textual instructions; MEM [38], which combines short-term video memory with long-term language memory for progress tracking; and Mem-0 [9], which uses a VLM planner to generate subtasks for a diffusion-based executor. While effective, these methods rely on additional language-reasoning modules, increasing parameters and computation.

**External Memory Modules in VLAs.** To avoid the overhead of high-level memory or planning modules, recent works introduce external memory modules into VLA architectures [33, 22, 30, 13], mainly through explicit memory-bank retrieval or compressed history modeling. In the first direction, MemoryVLA [33] stores visual details and semantic tokens in a perceptual-cognitive memory bank, while MAP-VLA [22] builds a demonstration memory library with task-stage soft prompts. However, such methods often store selective information, rely on reliable memory construction and subtask segmentation, and may suffer from ambiguous boundaries or accumulated retrieval errors. Compressed history methods summarize past observations into compact representations. HAMLET [19] encodes timesteps as moment tokens, ReMem-VLA [20] uses recurrent memory queries across frames and chunks, and MEM [38] compresses frame histories with a video encoder. Although these methods reduce the cost of full-history attention, they require architectural changes or dedicated design, limiting compatibility with pretrained VLAs and adding overhead as horizons grow. In contrast, NATIVEMEM encodes history using the pretrained VLA's vision encoder, and reuses the policy's existing attention with minimal overhead.

## III. METHOD

### A. Problem Formulation

We consider robotic manipulation in a partially observable environment. At control step $t$, a pretrained single-frame VLA policy maps the current multi-view observation $\mathbf{o}_t = \{o_t^v\}_{v=1}^V$ (with $V$ camera views), proprioceptive state $s_t$, and language instruction $\ell$ to actions,

$$\mathbf{a}_t \sim \pi_\theta(\cdot \mid \mathbf{o}_t, s_t, \ell), \tag{1}$$

where $\theta$ denotes the parameters of the pretrained VLA. However, such reactive policies are fundamentally limited for long-horizon manipulation, which is often non-Markovian: the correct action depends not only on the current input but on the interaction history $\mathcal{H}_t$ available before step $t$. Crucially, identical current observations can demand different actions depending on what happened before. Formally, there may exist two histories $\mathcal{H}_t \neq \mathcal{H}'_t$ such that

$$\begin{aligned} &\mathbf{o}_t = \mathbf{o}'_t, \quad s_t = s'_t, \quad \ell = \ell', \\ &\text{but} \quad \mathbf{a}_t^\star(\mathcal{H}_t, \mathbf{o}_t, s_t, \ell) \neq \mathbf{a}_t^\star(\mathcal{H}'_t, \mathbf{o}_t, s_t, \ell). \end{aligned} \tag{2}$$

This is the case whenever the robot must recall previously manipulated objects, operation counts, intermediate progress, or failed attempts. In all of these, the missing ingredient is historical information, and a policy that ignores it is underspecified by construction. We therefore convert a pretrained single-frame VLA into a memory-augmented policy that additionally conditions on this history,

$$\mathbf{a}_t \sim \pi_{\theta'}(\cdot \mid \mathbf{o}_t, s_t, \ell, \mathcal{M}_t), \tag{3}$$

where $\mathcal{M}_t$ is a compact memory built from the history $\mathcal{H}_t$ and $\theta'$ are the adapted parameters. The central challenge is to design $\mathcal{M}_t$ to preserve action-relevant history while staying efficient and compatible with the pretrained VLA, motivating our native memory compression scheme.

### B. Native Memory Compression via Action Supervision

Consider a policy that maintains a visual history over $T$ seconds and updates memory at $N$ frames per second. If each historical frame from each camera view is represented by $M$ tokens, the memory length grows as

$$|\mathcal{M}_t| = V \cdot T \cdot N \cdot M, \tag{4}$$

where $V$ is the number of camera views. This scaling reveals a fundamental **memory fidelity and efficiency trade-off**: increasing the temporal horizon $T$ or the update frequency $N$ improves the fidelity of historical context, but rapidly expands the VLA input sequence and its associated computation. Conversely, reducing the memory length improves efficiency, but risks discarding action-critical details. To retain fine-grained histories without overwhelming the VLA context, the per-frame token cost $M$ must therefore be aggressively compressed.

Our key idea, **Native Memory Compression**, is to push this compression to an extreme by setting $M = 1$: each frame-view pair is summarized into a single memory token. However, compactness alone is insufficient. Since the memory tokens are consumed by a pretrained VLA, they must also be compatible with its native token space and action-generation prior. Arbitrary latent states from external modules may not be interpretable by the VLA, while summaries optimized for reconstruction or generic visual representation learning are not necessarily useful for action prediction. We therefore learn compact memory tokens directly through VLA's original action objective.

To instantiate this idea, we derive a memory tokenizer from the VLA's own visual encoder, as shown in Fig. 2. Given a frame-view observation $o_\tau^v$, we initialize the memory encoder $E_{\text{mem}}$ from the pretrained VLA visual encoder and introduce a learnable memory query token $q_{\text{mem}}$. The query token attends to the visual patch-token sequence $\mathbf{P}_\tau^v$ and aggregates it into a single summary token:

$$[\hat{q}_\tau^v, \hat{\mathbf{P}}_\tau^v] = E_{\text{mem}}\left([q_{\text{mem}}, \mathbf{P}_\tau^v]\right), \tag{5}$$

where $\hat{q}_\tau^v$ is the output memory summary. A linear memory projection maps this summary into the VLA token dimension by

$$m_\tau^v = W_{\text{mem}} \hat{q}_\tau^v. \tag{6}$$

The resulting $m_\tau^v$ can serve as a single native visual memory token for frame $\tau$ and view $v$.

Given a set of memory frame indices $\mathcal{I}_t$, we concatenate memory tokens across time and views, preceded by a learnable memory beginning-of-sequence token $b_{\text{mem}}$:

$$\mathcal{M}_t = [b_{\text{mem}}, \{m_\tau^v \mid \tau \in \mathcal{I}_t,\ \tau \leq t,\ v = 1, \ldots, V\}]. \tag{7}$$

The memory sequence is appended to the original VLA input sequence,

$$\mathbf{x}_t = \left[\mathbf{x}_t^{\text{obs}}, \mathbf{x}^{\text{prompt}}(\ell, s_t), \mathcal{M}_t\right], \tag{8}$$

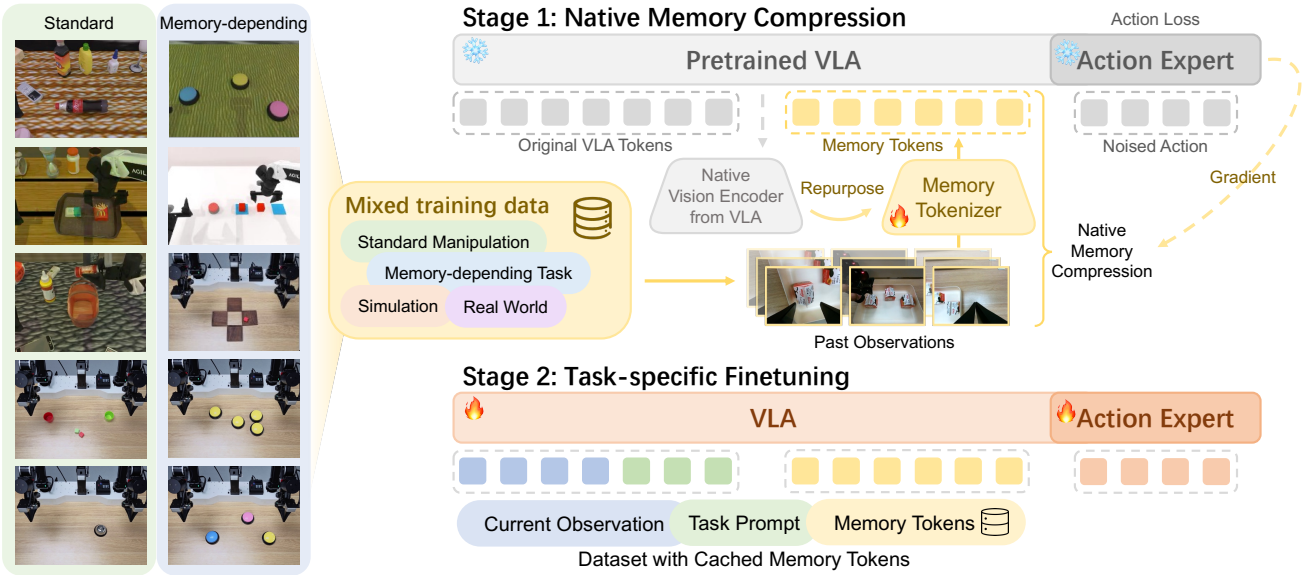



Fig. 2: **Training pipeline of NativeMEM. Stage 1:** We freeze the pretrained VLA and learn a native memory tokenizer, initialized from its visual encoder, using the VLA's original action prediction loss on mixed standard and memory-dependent manipulation data. The tokenizer compresses each frame-view observation into a single action-aligned memory token. **Stage 2:** We cache memory tokens and finetune the VLA with the original action loss by appending compressed memory tokens to the standard current-observation and prompt tokens.

where $\mathbf{x}_t^{\text{obs}}$ denotes the current observation tokens and $\mathbf{x}^{\text{prompt}}(\ell, s_t)$ denotes the language and proprioceptive conditioning tokens. This introduces historical context without changing the VLA architecture: the policy attends to memory through its existing token-processing pipeline.

We learn these native memory tokens in a first-stage alignment procedure. The pretrained VLA is frozen, and only the memory branch is trainable. Given the augmented token sequence $\mathbf{x}_t$, the frozen VLA predicts the target action using its original action head, and the memory branch is optimized by the native VLA action loss:

$$\min_{\phi_{\text{mem}}, q_{\text{mem}}, W_{\text{mem}}, b_{\text{mem}}} \mathcal{L}_{\text{act}}\Big(\pi_\theta(\cdot \mid \mathbf{o}_t, s_t, \ell, \mathcal{M}_t), \mathbf{a}_t\Big), \quad (9)$$

where $\theta$ is fixed and $\phi_{\text{mem}}$ denotes the parameters of the memory encoder. Since gradients can only update the memory branch, the learned tokens are encouraged to encode information that is both action-relevant and aligned with the frozen VLA's pretrained token space.

For training efficiency, we do not load the full visual history. Instead, for each training step, $\mathcal{I}_t$ consists of the first frame of the episode and a recent history window. The first frame provides coarse task initialization context, while the recent window captures state changes and progress. We also include the current frame in $\mathcal{I}_t$, so that the tokenizer learns a unified frame-level summarization behavior for both current and past observations.

We train the memory tokenizer on a mixture of standard manipulation data and memory-demanding tasks from both simulation and real-world robot demonstrations. Standard manipulation data preserves compatibility with the pretrained VLA's general manipulation prior, while memory-demanding tasks encourage the tokenizer to encode information that is useful beyond the current observation. The result is a general action-supervised native memory tokenizer that can be reused for downstream memory-augmented finetuning.

### C. Task-Specific Finetuning and Real-Time Memory Inference

**Task-specific finetuning.** Given a target-task demonstration dataset, we first use the learned memory tokenizer to preprocess the visual history offline. For each episode, every frame from each camera view is converted into its corresponding memory summary token $\hat{q}_\tau^v$. Since each frame-view pair is represented by only one token, this preprocessing introduces negligible storage and I/O overhead compared with storing dense visual token sequences.

For each training step $t$, we retrieve the cached summary tokens from selected frame indices $\mathcal{I}_t$ with $\tau \leq t$, and form the memory sequence following Eq. 7. The resulting memory sequence is appended to the standard VLA input as in Eq. 8.

We initialize the VLA backbone from the pretrained single-frame policy and load $W_{\text{mem}}$ and $b_{\text{mem}}$ from the first-stage memory branch. The memory tokenizer encoder is kept fixed during this stage, while the VLA backbone, action head, and memory projection are finetuned on limited target-task demonstrations using the native action prediction loss:

$$\min_{\theta', W_{\text{mem}}, b_{\text{mem}}} \mathcal{L}_{\text{act}}\left(\pi_{\theta'}(\cdot \mid \mathbf{o}_t, s_t, \ell, \mathcal{M}_t), \mathbf{a}_t\right). \quad (10)$$

Here, $\theta'$ denotes the finetuned VLA parameters. This preserves the standard VLA finetuning workflow while extending the policy with compact historical context.

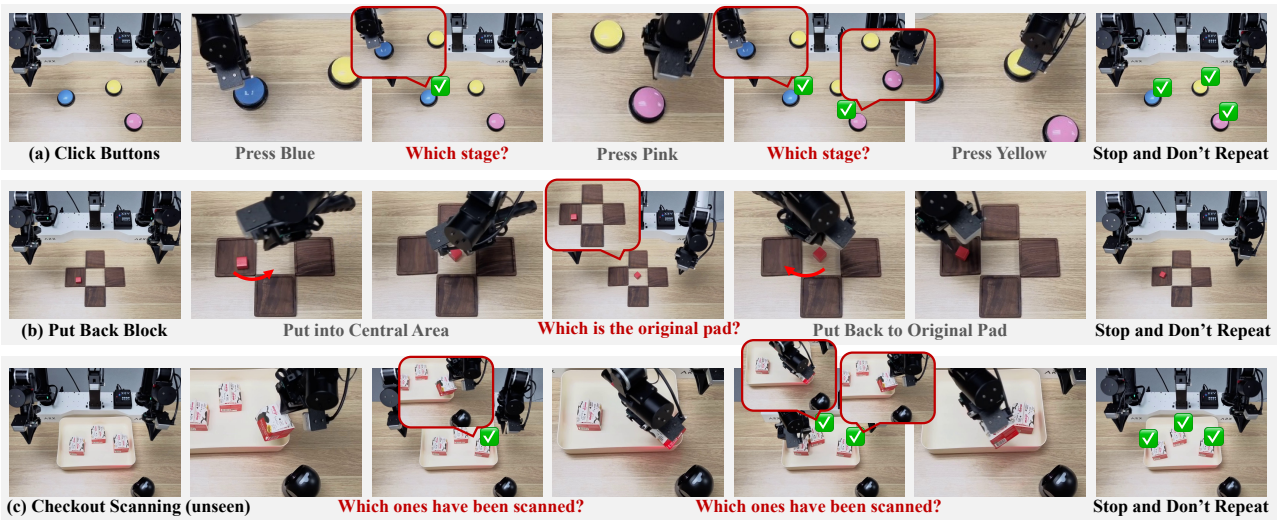



Fig. 3: **Memory-dependent manipulation tasks.** In *Click Buttons*, the robot must follow a specified button sequence without repeating completed presses. In *Put Back Block*, it must remember the block's original pad after moving it to the center. In *Grocery Checkout Scanning*, it must track which items have already been scanned and avoid duplicate or missed scans.

TABLE I: Task success rates (%) in simulation evaluation. The benchmark includes three RMBench tasks and two additional simulated button-pressing tasks. NATIVEMEM achieves the best performance across all tasks. The best result for each task is marked in **bold**. * indicates reproduced baselines implemented on the same $\pi_{0.5}$ backbone.

| Method | Click Buttons | Click Buttons (hard) | Swap Blocks | Put Back Block | Observe and Pickup | Avg. |
|---|---|---|---|---|---|---|
| *Base VLA Policies* | | | | | | |
| $\pi_{0.5}$ [15] | 0 | 0 | 24 | 11 | 9 | 8.8 |
| X-VLA [51] | 7 | 12 | 16 | 18 | 9 | 12.4 |
| *VLM-Driven Memory* | | | | | | |
| MemER [34] | 12 | 8 | 18 | 12 | 2 | 10.4 |
| Mem-0 [9] | 0 | 1 | 67 | 90 | 4 | 32.4 |
| *External Memory Modules* | | | | | | |
| HAMLET* [19] | 4 | 17 | 11 | 3 | 10 | 9.0 |
| MEM-short* [38] | 0 | 39 | 4 | 15 | 6 | 12.8 |
| **Ours** | **94** | **88** | **94** | **100** | **44** | **84.0** |

**Real-time memory inference.** During deployment, the memory tokenizer can operate independently from the VLA policy. As new observations arrive, the tokenizer converts them into memory tokens at a specified update frequency and maintains a compact memory queue. Since the tokenizer is derived from the original visual encoder and produces only one token per frame-view pair, memory updates can be performed with low overhead and do not require modifying the VLA inference pipeline. When the VLA is queried for an action, the current observation, proprioceptive state, and language instruction are processed as usual. The current memory queue is simply concatenated after the original input tokens, following Eq 8. Thus, memory construction and policy inference are decoupled: the tokenizer can update historical memory at a high frequency, while the VLA consumes the latest compact memory sequence whenever action prediction is required.

## IV. EXPERIMENTS

### A. Experimental Setup

**Datasets and Tasks.** We evaluate NATIVEMEM on memory-dependent manipulation tasks in both simulation and the real world. For real-world evaluation, we consider three tasks, as illustrated in Fig. 3. Notably, *Grocery Checkout Scanning (unseen)* is not included during first-stage memory-tokenizer training. This setting evaluates whether the learned native memory representation transfers to new forms of long-horizon task progress tracking. In simulation, we use three RMBench tasks [9] and two additional button-pressing tasks. The simulated *Click Buttons* task follows the same memory requirement as its real-world counterpart, while *Click Buttons (Hard)* removes color distinctions between buttons, forcing the policy to rely on spatial memory and interaction history.

**Baselines.** We compare NATIVEMEM with representative VLA policies that instantiate different memory-modeling

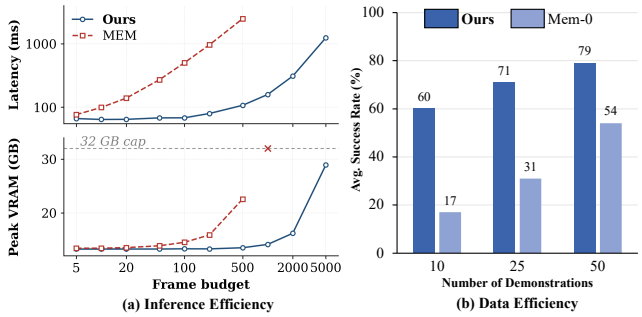



Fig. 4: (a) Inference latency and memory consumption. (b) Average data efficiency on three RMBench tasks.

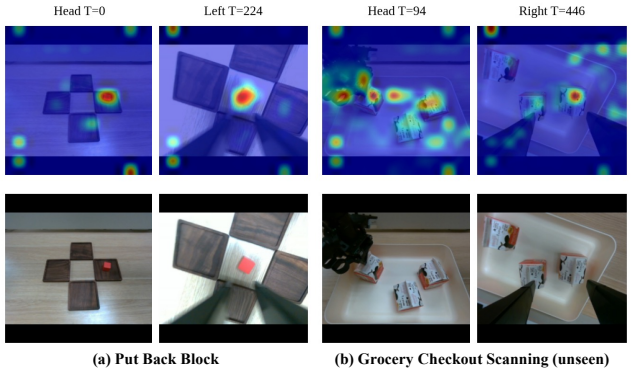



Fig. 5: Visualization of the spatial attention used by the memory tokenizer when compressing each frame-view observation into a single memory token. The top row shows the tokenizer attention maps over input images, while the bottom row shows the corresponding raw observations.

paradigms. For *VLM-driven memory*, we include MemER [34] and Mem-0 [9]. For *external memory modules*, we include HAMLET [19] and MEM-short [38]. For a fair comparison, we implement both HAMLET and MEM-short on the same $\pi_{0.5}$ backbone used by NATIVEMEM. Since HAMLET and MEM-short are not publicly released, we faithfully reproduce them following the official papers. MEM-short is originally designed to be retrained on large-scale data. To ensure a controlled comparison, we train it using the same demonstration trajectories available to all methods. Therefore mark them as HAMLET* and MEM-short* in Tab. I and II.

### B. Main Results

**Simulation Experiments.** As shown in Tab. I, NATIVEMEM achieves the best performance across all tasks, improving the average success rate to $84.0\%$. The evaluated tasks cover two complementary memory requirements: long-range recall of early observations, as in *Put Back Block* and *Observe and Pickup*, and continuous progress tracking, as in *Click Buttons*, *Click Buttons (Hard)*, and *Swap Blocks*. Existing VLM-driven or short-horizon memory methods struggle with one or both settings, while NATIVEMEM remains consistently effective, suggesting its native memory compression provides a unified representation for both recalling visual evidence and tracking task states.

**Real-World Experiments.** Tab. II shows that NATIVEMEM also transfers effectively to real-robot manipulation, achieving the highest overall success rate across all tasks. While RTC improves over the $\pi_{0.5}$ policy by exploiting temporal action continuity, rather than explicitly remembering past observations. Its performance still drops in later stages that require explicit historical recall. MEM-short* becomes less stable under limited real-world data, occasionally degrading the pretrained VLA's manipulation capability. In contrast, NATIVEMEM preserves the pretrained policy prior while adding compact history, achieving the strongest performance across all three tasks.

**Inference Efficiency.** Fig. 4(a) compares inference latency and peak GPU memory under increasing history length. Our NATIVEMEM supports up to 5,000 historical frames within a 32 GB memory budget, and still attends to about 200 frames under a 100 ms real-time latency constraint, enabling substantially longer histories while preserving the real-time reactiveness.

**Data-Efficiency.** Fig. 4(b) shows that NATIVEMEM adapts effectively with limited demonstrations. With only 10 demonstrations, it reaches $60\%$ average success, $3.5\times$ higher than Mem-0, and consistently outperforms Mem-0 with 25 and 50 demonstrations, reflecting the benefit of reusing pretrained VLA manipulation priors.

### C. Qualitative Analysis: What Does Native Memory Capture?

**Spatial Attention of the Memory Tokenizer.** To understand what is encoded into each memory token, we visualize the tokenizer's spatial attention over historical observations. As shown in Fig. 5, the tokenizer consistently focuses on manipulation-relevant regions rather than background pixels. For *Put Back Blocks*, attention concentrates on the block and its corresponding pad. Notably, this behavior also generalizes to *Grocery Checkout Scanning*, which is unseen during tokenizer training: the tokenizer still attends to foreground objects, especially the item about to be grasped.

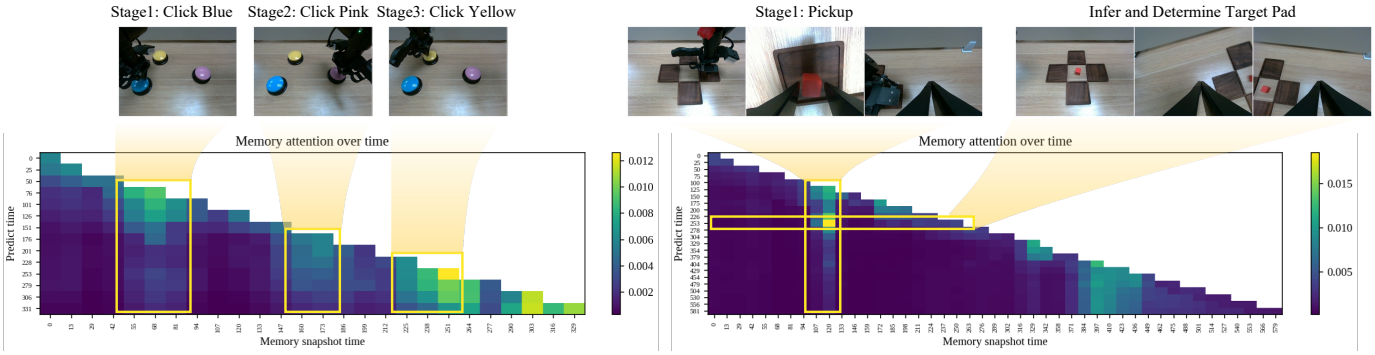
**Action Attention over the Memory Sequence.** To examine how NATIVEMEM uses history during action generation, we visualize the attention from action tokens to memory tokens across inference time. Fig. 6 shows that the policy attends to task-relevant moments rather than simply the most recent observations. In *Click Buttons*, high-attention memories align with the three timesteps when individual buttons were pressed. In *Put Back Block*, attention concentrates on the moment when the block was lifted from its original pad, which is critical for deciding where to return it.

### D. Ablation Study

**Native Memory Compression.** As shown in Tab. III (lines 1 and 2), when the VLA is unfrozen during memory-compression learning, performance drops to $23.2\%$. The model reduces the action loss by directly adapting the pretrained VLA itself rather than forcing the memory branch. After

TABLE II: Real-world cumulative task success rates (%). S1-S3 denote cumulative success after each task stage; the last stage corresponds to the overall task success rate. *Grocery Checkout Scanning* is unseen during first-stage memory-tokenizer training. The mean overall success rate is reported. The best results are marked in **bold**. * denotes reproduced baselines trained with the same number of task-specific demonstrations as NATIVEMEM.

| Method | Click Buttons | | | Put Back Block | | Grocery Checkout Scanning (unseen) | | | Avg. |
|---|---|---|---|---|---|---|---|---|---|
| | S1 | S2 | S3 | S1 | S2 | S1 | S2 | S3 | |
| $\pi_{0.5}$ [15] | 16 | 8 | 2 | 72 | 14 | 90 | 58 | 28 | 14.7 |
| $\pi_{0.5}$ + RTC [4] | 26 | 18 | 16 | 56 | 24 | 74 | 66 | 64 | 34.7 |
| Mem-0 [9] | 36 | 4 | 0 | 0 | 0 | 6 | 0 | 0 | 0.0 |
| MEM-short* [38] | 72 | 52 | 40 | 6 | 0 | 72 | 56 | 52 | 30.7 |
| **Ours** | **100** | **98** | **96** | **100** | **100** | **100** | **100** | **100** | **98.7** |




(a) Click Buttons (b) Put Back Block

Fig. 6: Visualization of attention from action tokens to historical memory snapshots. The x-axis denotes the timestamp of each memory snapshot, and the y-axis denotes the action prediction timestep. Each row shows which memories are used for a given action prediction time, while each column indicates how a specific memory is attended by future predictions. The attention in *Click Buttons* follows the three clicking stages, while in *Put Back Block* it concentrates on the pickup moment needed to infer the target pad.

| Method | Click Button | Click Button (hard) | Swap Blocks | Put Back Block | Observe and Pickup | Avg. |
|---|---|---|---|---|---|---|
| Unfrozen VLA | 38 | 24 | 45 | 0 | 9 | 23.2 |
| w/o Stage1 | 94 | 80 | 92 | 17 | 7 | 58.0 |
| Sparse update (0.5Hz) | 83 | 63 | 91 | 24 | 9 | 53.8 |
| Short horizon (2s) | 26 | 87 | 0 | 18 | 39 | 34.0 |
| Short horizon (4s) | 69 | 87 | 0 | 18 | 37 | 43.0 |
| Short horizon (6s) | 80 | 88 | 0 | 18 | 44 | 46.0 |
| **Ours** | **94** | **88** | **94** | **100** | **44** | **84.0** |

TABLE III: **Ablation Study on Native Memory Compression and its Temporal Coverage.** The first two variants evaluate the memory-compression learning, while the remaining variants examine the importance of long-horizon and fine-grained history. The best results are marked in **bold**.

skipping Native Memory Compression, mean-pooled vision-encoder features are used as memory tokens. The results show that such generic visual features can capture coarse historical context, but fail to preserve details required by tasks such as *Put Back Block* and *Observe and Pickup*. These results indicate that Native Memory Compression is essential for distilling task-relevant historical cues, enabling the policy to recover critical information needed for downstream action prediction.

**Temporal Memory Coverage.** Sparse update at 0.5Hz broadly reduces success rates, showing that the policy needs fine-grained temporal evidence to track interaction states. Short horizons degrade tasks whose critical information falls outside the retained window, such as *Swap Blocks* and *Put Back Block*. Results in Tab. III (lines 4∼6) show that NATIVEMEM's performance relies on both dense temporal coverage and long-horizon retention: it updates memory frequently enough to capture fine-grained interaction changes, while its compact native tokens make minute-level history retention scalable.

## V. CONCLUSION

We presented NATIVEMEM, which enables pretrained single-frame VLAs to retain long-horizon, fine-grained vi-

sual histories through **Native Memory Compression**. By repurposing the VLA's own vision encoder, NATIVEMEM achieves one-token-per-frame compression. We further introduced a two-stage training pipeline that first learns an action-supervised memory tokenizer aligned with the pretrained VLA's visual-action priors, and then performs task-specific finetuning with limited demonstrations. Across simulation and real-world manipulation tasks, NATIVEMEM substantially improves average success rates from 32.4% to 84.0% in simulation and from 34.7% to 98.7% on real robots, while matching leading memory-designed methods with only 20% of the training data.

## REFERENCES


[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[2] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. PaliGemma: A versatile 3b VLM for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

[3] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. $\pi_0$: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[4] Kevin Black, Allen Z Ren, Michael Equi, and Sergey Levine. Training-time action conditioning for efficient real-time chunking. *arXiv preprint arXiv:2512.05964*, 2025.

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

[6] 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.

[7] 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.

[8] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. SpatialVLM: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[9] Tianxing Chen, Yuran Wang, Mingleyang Li, Yan Qin, Hao Shi, Zixuan Li, Yifan Hu, Yingsheng Zhang, Kaixuan Wang, Yue Chen, et al. RMBench: Memory-Dependent robotic manipulation benchmark with insights into policy design. *arXiv preprint arXiv:2603.01229*, 2026.

[10] Zhiwei Chen, Yupeng Hu, Zhiheng Fu, Zixu Li, Jiale Huang, Qinlei Huang, and Yinwei Wei. INTENT: Invariance and discrimination-aware noise mitigation for robust composed image retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

[11] Haoquan Fang, Markus Grotz, Wilbert Pumacay, Yi Ru Wang, Dieter Fox, Ranjay Krishna, and Jiafei Duan. SAM2Act: Integrating visual foundation model with a memory architecture for robotic manipulation. *arXiv preprint arXiv:2501.18564*, 2025.

[12] Yihuai Gao, Jinyun Liu, Shuang Li, and Shuran Song. Gated memory policy. *arXiv preprint arXiv:2604.18933*, 2026.

[13] Xinying Guo, Chenxi Jiang, Hyun Bin Kim, Ying Sun, Yang Xiao, Yuhang Han, and Jianfei Yang. Chameleon: Episodic memory for long-horizon robotic manipulation. *arXiv preprint arXiv:2603.24576*, 2026.

[14] Yutong Hu, Jan-Nico Zaech, Nikolay Nikolov, Yuanqi Yao, Sombit Dey, Giuliano Albanese, Renaud Detry, Luc Van Gool, and Danda Paudel. AR-VLA: True autoregressive action expert for Vision-Language-Action models. *arXiv preprint arXiv:2603.10126*, 2026.

[15] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with Open-World generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[16] 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.

[17] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

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

[19] 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.

[20] 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.

[21] Haoyun Li, Ivan Zhang, Runqi Ouyang, Xiaofeng Wang, Zheng Zhu, Zhiqin Yang, Zhentao Zhang, Boyuan Wang, Chaojun Ni, Wenkang Qin, et al. MimicDreamer: Aligning human and robot demonstrations for scalable VLA

training. *arXiv preprint arXiv:2509.22199*, 2025.

[22] Runhao Li, Wenkai Guo, Zhenyu Wu, Changyuan Wang, Haoyuan Deng, Zhenyu Weng, Yap-Peng Tan, and Ziwei Wang. MAP-VLA: Memory-augmented prompting for vision-language-action model in robotic manipulation. *arXiv preprint arXiv:2511.09516*, 2025.

[23] Yongkang Li, Lijun Zhou, Sixu Yan, Bencheng Liao, Tianyi Yan, Kaixin Xiong, Long Chen, Hongwei Xie, Bing Wang, Guang Chen, et al. UniDriveVLA: Unifying understanding, perception, and action planning for autonomous driving. *arXiv preprint arXiv:2604.02190*, 2026.

[24] Zixu Li, Yupeng Hu, Zhiwei Chen, Qinlei Huang, Guozhi Qiu, Zhiheng Fu, and Meng Liu. ReTrack: Evidence-driven dual-stream directional anchor calibration network for composed video retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

[25] Zixu Li, Yupeng Hu, Zhiwei Chen, Shiqi Zhang, Qinlei Huang, Zhiheng Fu, and Yinwei Wei. HABIT: Chrono-synergia robust progressive learning framework for composed image retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

[26] 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. *arXiv preprint arXiv:2512.09928*, 2025.

[27] Jiahang Liu, Yunpeng Qi, Jiazhao Zhang, Minghan Li, Shaoan Wang, Kui Wu, Hanjing Ye, Hong Zhang, Zhibo Chen, Fangwei Zhong, et al. TrackVLA++: Unleashing reasoning and memory capabilities in VLA models for embodied visual tracking. *arXiv preprint arXiv:2510.07134*, 2025.

[28] Andrés Marafioti, Orr Zohar, Miquel Farré, Merve Noyan, Elie Bakouch, Pedro Cuenca, Cyril Zakka, Loubna Ben Allal, Anton Lozhkov, Nouamane Tazi, et al. SmolVLM: Redefining small and efficient multimodal models. *arXiv preprint arXiv:2504.05299*, 2025.

[29] Geraud Nangue Tasse, Matthew Riemer, Benjamin Rosman, and Tim Klinger. Beyond sliding windows: Learning to manage memory in non-markovian environments. *arXiv e-prints*, 2025.

[30] Maëlic Neau, Zoe Falomir, Paulo E Santos, Anne-Gwenn Bosser, and Cédric Buche. GraSP-VLA: Graph-based symbolic action representation for long-horizon planning with VLA policies. *arXiv preprint arXiv:2511.04357*, 2025.

[31] Chaojun Ni, Cheng Chen, Xiaofeng Wang, Zheng Zhu, Wenzhao Zheng, Boyuan Wang, Tianrun Chen, Guosheng Zhao, Haoyun Li, Zhehao Dong, et al. SwiftVLA: Unlocking spatiotemporal dynamics for lightweight vla models at minimal overhead. *CVPR 2026*, 2026.

[32] Fabian Peller-Konrad, Rainer Kartmann, Christian RG Dreher, Andre Meixner, Fabian Reister, Markus Grotz, and Tamim Asfour. A memory system of a robot cognitive architecture and its implementation in ArmarX. *Robotics and Autonomous Systems*, 2023.

[33] 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. *arXiv preprint arXiv:2508.19236*, 2025.

[34] 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.

[35] GigaBrain Team, Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Haoyun Li, Jie Li, Jiagang Zhu, Lv Feng, et al. GigaBrain-0: A world model-powered vision-language-action model. *arXiv preprint arXiv:2510.19430*, 2025.

[36] GigaBrain Team, Boyuan Wang, Bohan Li, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. GigaBrain-0.5M$^*$: a vla that learns from world model-based reinforcement learning. *arXiv preprint arXiv:2602.12099*, 2026.

[37] GigaWorld Team, Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Haoyun Li, Jiagang Zhu, Kerui Li, Mengyuan Xu, et al. GigaWorld-0: World models as data engine to empower embodied ai. *arXiv preprint arXiv:2511.19861*, 2025.

[38] Marcel Torne, Karl Pertsch, Homer Walke, Kyle Vedder, Suraj Nair, Brian Ichter, Allen Z Ren, Haohuan Wang, Jiaming Tang, Kyle Stachowicz, et al. MEM: Multi-scale embodied memory for vision language action models. *arXiv preprint arXiv:2603.03596*, 2026.

[39] Honghui Wang, Zhi Jing, Jicong Ao, Shiji Song, Xuelong Li, Gao Huang, and Chenjia Bai. Beyond Short-Horizon: VQ-Memory for robust long-horizon manipulation in non-markovian simulation benchmarks. *arXiv preprint arXiv:2603.09513*, 2026.

[40] Yiru Wang, Zichong Gu, Yu Gao, Anqing Jiang, Zhigang Sun, Shuo Wang, Yuwen Heng, and Hao Sun. HiST-VLA: A hierarchical spatio-temporal vision-language-action model for end-to-end autonomous driving. *arXiv preprint arXiv:2602.13329*, 2026.

[41] Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Zhibin Tang, Kun Wu, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, et al. TinyVLA: Towards fast, data-efficient vision-language-action models for robotic manipulation. *IEEE Robotics and Automation Letters*, 2025.

[42] Jiazhi Yang, Kunyang Lin, Jinwei Li, Wencong Zhang, Tianwei Lin, Longyan Wu, Zhizhong Su, Hao Zhao, Ya-Qin Zhang, Li Chen, et al. RISE: Self-improving robot policy with compositional world model. *arXiv preprint arXiv:2602.11075*, 2026.

[43] Yuncong Yang, Han Yang, Jiachen Zhou, Peihao Chen, Hongxin Zhang, Yilun Du, and Chuang Gan. 3D-mem: 3D scene memory for embodied exploration and reasoning. In *Proceedings of the Computer Vision and*

*Pattern Recognition Conference*, 2025.

[44] Angen Ye, Zeyu Zhang, Boyuan Wang, Xiaofeng Wang, Dapeng Zhang, and Zheng Zhu. VLA-R1: Enhancing reasoning in vision-language-action models. *arXiv preprint arXiv:2510.01623*, 2025.

[45] Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. GigaWorld-Policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026.

[46] Zijian Zeng, Fei Ding, Huiming Yang, and Xianwei Li. HELM: Harness-Enhanced long-horizon memory for vision-language-action manipulation. *arXiv preprint arXiv:2604.18791*, 2026.

[47] Andy Zhai, Brae Liu, Bruno Fang, Chalse Cai, Ellie Ma, Ethan Yin, Hao Wang, Hugo Zhou, James Wang, Lights Shi, et al. Igniting VLMs toward the embodied space. *arXiv preprint arXiv:2509.11766*, 2025.

[48] Mingliang Zhai, Zhi Gao, Yuwei Wu, and Yunde Jia. Memory-Centric embodied question answering. *arXiv preprint arXiv:2505.13948*, 2025.

[49] 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. *arXiv preprint arXiv:2506.22242*, 2025.

[50] 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. *arXiv preprint arXiv:2507.04447*, 2025.

[51] Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025.

[52] Shijie Zhou, Alexander Vilesov, Xuehai He, Ziyu Wan, Shuwang Zhang, Aditya Nagachandra, Di Chang, Dongdong Chen, Xin Eric Wang, and Achuta Kadambi. VLM4D: Towards spatiotemporal awareness in vision language models. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2025.

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