2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

MAP-VLA: Memory-Augmented Prompting for Vision-Language-Action Model in Robotic Manipulation

Runhao Li[1], Wenkai Guo[1], Zhenyu Wu[3], Changyuan Wang[4], Haoyuan Deng[1], Zhenyu Weng[5], Yap-Peng Tan[2,1], Ziwei Wang[1,*]

***Abstract*— Pre-trained Vision-Language-Action (VLA) models have achieved remarkable success in improving robustness and generalization for end-to-end robotic manipulation. However, these models struggle with long-horizon tasks due to their lack of memory and reliance solely on immediate sensory inputs. To address this limitation, we propose Memory-Augmented Prompting for Vision-Language-Action model (MAP-VLA), a novel framework that empowers pre-trained VLA models with demonstration-derived memory prompts to augment action generation for long-horizon robotic manipulation tasks. To achieve this, MAP-VLA first constructs a memory library from historical demonstrations, where each memory unit captures information about a specific stage of a task. These memory units are implemented as learnable soft prompts optimized through prompt tuning. Then, during real-time task execution, MAP-VLA retrieves relevant memory through trajectory similarity matching and dynamically integrates it into the VLA model for augmented action generation. Importantly, this prompt tuning and retrieval augmentation approach operates as a plug-and-play module for a frozen VLA model, offering a lightweight and flexible solution to improve task performance. Experimental results show that MAP-VLA delivers up to 7.0% absolute performance gains in the simulation benchmark and 25.0% on real robot evaluations for long-horizon tasks, surpassing the current state-of-the-art methods.**

I. INTRODUCTION

Robotic manipulation remains a long-standing challenge in embodied artificial intelligence, requiring a robot to perceive complex scenes, understand task goals, and generate multi-step control actions. Traditional approaches [1] often rely on task-specific pipelines or require substantial engineering and manual tuning, which limits their adaptability and scalability. In response to these limitations, Vision-Language-Action (VLA) models [2], [3], [4], [5], [6] have emerged as a compelling new paradigm for developing generalist robotic manipulation policies. These models build upon the success of large-scale vision-language foundation models by extending their capabilities into the action domain. VLA models acquire broad knowledge about the world from vision-language pre-training and learn to map visual observations and natural language instructions directly to robot actions through end-to-end training on diverse robotic datasets [7], [8]. By unifying visual perception, language understanding, and action generation into a single policy, they offer a powerful foundation for general-purpose robotic manipulation.

Despite the advantages mentioned above, current VLA models have a key limitation: they fail to leverage historical memory at task execution. Once a VLA model is trained, it relies solely on immediate sensory inputs to decide the subsequent actions. The rich experience contained in the training demonstrations is not explicitly accessible during execution as historical memory, and is only used to train the model parameters offline. In other words, the robot lacks episodic memory; it cannot directly recall "how an expert accomplished a similar stage" when it is performing one. This stateless execution is suboptimal, especially for long-horizon tasks, and the robot may deviate from the intended trajectory in challenging situations that an expert has encountered before. Existing approaches have explored in-context learning for robots [9], [10], [11], [12], for instance, by providing a few demonstration episodes as part of the input to a policy model. However, such solutions either require extensive training tailored to specific architectures and input configurations, or concentrate on LLMs without applicability to pre-trained VLA models. In general, existing VLA models do not natively support the use of historical data for action generation at test time, leaving a gap between how humans solve long-horizon tasks, by recalling past memory, and how these models operate. This motivates the need for approaches that can endow VLA models with a lightweight form of episodic memory.

In this paper, we present the Memory-Augmented Prompting for Vision-Language-Action model (MAP-VLA), bridging the gap in current VLA models by enabling dynamic access to demonstration-derived memory. As illustrated in Fig. 1, our insight is to equip a frozen VLA model with a lightweight memory of training demonstrations, which can be retrieved during task execution to provide stage-by-stage guidance. To achieve this, we first introduce Memory Prompt Construction, where we segment training trajectories into distinct stages and encode each stage's memory into a soft prompt via prompt tuning. This process effectively builds an external memory library that captures stage-specific knowledge for guiding the model's behavior. Next, we propose Memory-Augmented Action Generation, which retrieves the most relevant stage-specific memory prompt along with the corresponding demonstration actions by comparing the

This research is supported in part by the NTU Start up Grant (024303-00001), the Singapore National Robotics Programme Research Project DS-RFM (M25N4N2009), and the National Research Foundation, Singapore, under the NRF Medium Sized Centre Scheme (CARTIN). Any opinions, findings and conclusions expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore.

[1]Nanyang Technological University, Singapore, Singapore
[2]VinUniversity, Hanoi, Vietnam
[3]Beijing University of Posts and Telecommunications, Beijing, China
[4]Tsinghua University, Beijing, China
[5]South China University of Technology, Guangzhou, China
[*]Corresponding author: ziwei.wang@ntu.edu.sg

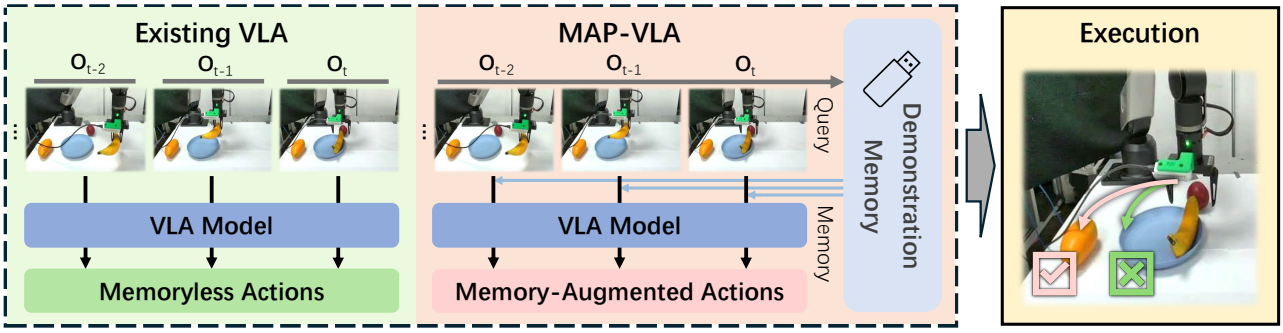

Fig. 1. Simplified execution pipeline of existing VLA methods and MAP-VLA.

trajectory similarity. The memory prompt and demonstration actions are subsequently utilized for prompt ensembling, which dynamically balances the advantages of the stage-specific memory prompts and the generalized base prompts. This whole framework, shown in Fig. 2, remains lightweight and flexible without updating the underlying model parameters. Extensive experiments demonstrate that MAP-VLA surpasses the state-of-the-art methods on long-horizon tasks with up to 7.0% absolute gains in the simulation benchmark and 25.0% in real robot evaluations. The main contributions of this work can be summarized as follows:

- We propose MAP-VLA, a novel framework that augments a pre-trained VLA model with demonstration-derived memory prompts. This framework operates on prompt tuning and retrieval-augmented generation to improve task adaptation, without requiring any modification to the model's internal weights.
- We introduce Memory Prompt Construction (MPC), which encodes stage-specific memory from expert demonstrations into a library of prompts to provide episodic task memory. We also develop Memory-Augmented Action Generation (MAAG), which enables memory retrieval and dynamic memory-aware prompt ensembling to augment action generation during real-time task execution.
- Extensive experiments show that MAP-VLA outperforms the state-of-the-art methods for long-horizon tasks, achieving up to 7.0% absolute gains in the simulation benchmark and 25.0% in real robot evaluations.

II. RELATED WORK

A. Vision-Language-Action Models

Recent advances in robot learning have introduced VLA models [2], [3], [4], [5], [6] pre-trained on large datasets [7], [8] that unify visual perception, natural language, and robot control. A prominent example is RT-2 (Robotic Transformer 2) [2], which introduced the VLA paradigm by integrating an Internet-scale vision-language model into an end-to-end policy. RT-2 represents robot actions as textual tokens and co-trains on both web data and robot demonstrations, yielding a single model that maps images to actions while enjoying the semantic richness of language pre-training. Building on this idea, OpenVLA [3] is a 7B-parameter open-source VLA model trained on nearly one million real-world robot episodes. It demonstrated efficient fine-tuning for new tasks and embodiments, significantly outperforming prior imitation learning methods (e.g. diffusion policies) on multi-task benchmarks. The π_0 model [4] proposes a novel flow-matching VLA architecture built on top of a pre-trained vision-language model. It employs a diffusion-like flow matching strategy for action generation, enabling controlling high-frequency, highly dexterous skills (up to 50Hz control rates) that are challenging for earlier autoregressive Transformers. However, despite differences in architecture and training, these methods share a common limitation: they rely solely on immediate sensory inputs and predefined task instructions, without utilizing the rich historical memory embedded in expert demonstrations during task execution.

B. Prompt Tuning

As foundation models became prevalent, prompt tuning [13], [14], [15], [16], [17] emerged as a lightweight method to adapt them to downstream tasks without full fine-tuning. In natural language processing, the literature [13] introduced prompt tuning as learning a set of continuous "soft prompt" vectors that are prepended to the input, conditioning a frozen language model to perform a new task. Instead of hand-crafting a textual prompt or updating millions of model weights, prompt tuning optimizes a small embedding that steers the model's behavior via its inputs. These soft prompts are learned through standard backpropagation on the task objective, effectively encoding the task information in the input space. Remarkably, this approach can achieve performance on par with fine-tuning the entire model, especially as model size grows. Prompt-based adaptation has also proven effective in vision-language models [14], [15], [16]. For instance, CoOp [14] applies soft prompt tuning to CLIP [18], a vision-language model, for image recognition tasks. CoOp replaces manual prompt engineering (e.g., finding the right wording like "a photo of a [class]") with learnable prompt embeddings, while freezing the entire CLIP model. Even with only a few training examples, the learned prompts can significantly improve accuracy over handcrafted prompts, and they transfer well to new domains. More broadly, prompt tuning is well-suited for VLA models due to their structural similarity with large language and vision-language models.

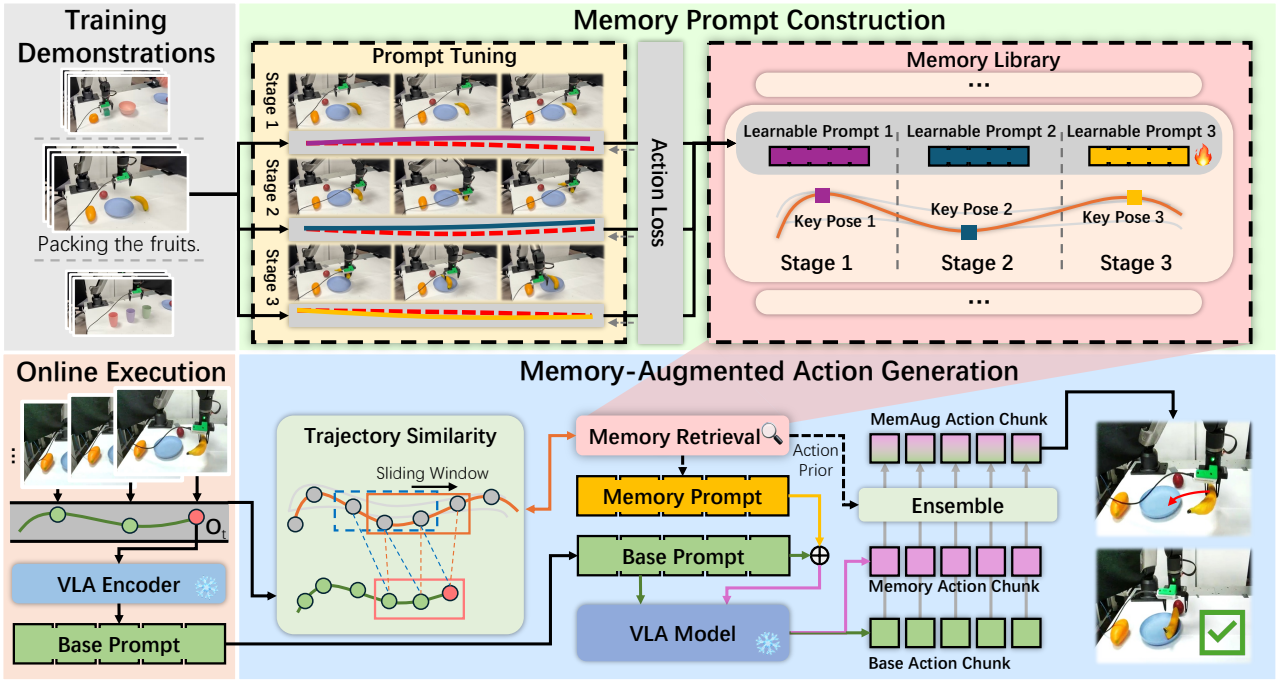

Fig. 2. The framework of MAP-VLA. Our method augments a frozen pre-trained VLA model with demonstration-derived memory prompts for enhanced action generation during task execution. The Memory Prompt Construction stage encodes stage-specific knowledge from expert demonstrations into a library of memory prompts. The Memory-Augmented Action Generation stage retrieves the memory prompts and augments action generation with memory-aware prompt ensembling.

III. METHODOLOGY

A. Preliminaries and Overview

Vision-Language-Action pre-training. VLA models are policies pre-trained on large-scale demonstrations to map multi-modal observations to robot actions. Each demonstration consists of a sequence of observation-action pairs $\{\mathbf{o}_t, \mathbf{a}_t\}_{t=1}^{n}$, where each observation $\mathbf{o}_t = [\mathbf{I}_t^1, \mathbf{I}_t^2, \ell_t, \mathbf{s}_t]$ includes an overview image $\mathbf{I}_t^1$, a wrist image $\mathbf{I}_t^2$, a language token sequence ℓ_t, and the robot state $\mathbf{s}_t$. The corresponding action at time t is represented by $\mathbf{a}_t$. Our method is based on the pre-trained π_0 [4] which uses $\mathbf{o}_t$ to predict an action chunk of H future actions $\mathbf{A}_t = [\mathbf{a}_t, \ldots, \mathbf{a}_{t+H-1}]$ via a conditional flow matching process. The learning objective can be summarized as follows:

$$L^\tau(\theta) = \mathbb{E}_{p(\mathbf{A}_t|\mathbf{o}_t),\, q(\mathbf{A}_t^\tau|\mathbf{A}_t)} \left\| \mathbf{f}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t) - \mathbf{u}(\mathbf{A}_t^\tau|\mathbf{A}_t) \right\|^2, \tag{1}$$

where $\tau \in [0, 1]$ is the flow matching timestep, and $\mathbf{A}_t^\tau = \tau \mathbf{A}_t + (1-\tau)\epsilon$ is a noisy interpolation with $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. The model $\mathbf{f}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t)$ predicts the denoising vector field $\mathbf{u}(\mathbf{A}_t^\tau|\mathbf{A}_t) = \epsilon - \mathbf{A}_t$. While π_0 and other VLA models excel at learning generalizable mappings from diverse demonstrations, they often struggle in long-horizon tasks due to the absence of memory mechanisms.

Methodology overview. To overcome this, we introduce a memory-augmented framework that enhances VLA models for better long-horizon task performance. Our framework comprises two components: (i) Memory Prompt Construction (MPC), which encodes stage-level memory into soft prompts derived from expert demonstrations, and (ii) Memory-Augmented Action Generation (MAAG), which retrieves and dynamically integrates these prompts to augment real-time action generation. We detail each component in Section III-B and Section III-C, respectively.

B. Memory Prompt Construction

Stage segmentation and alignment. We begin by partitioning each demonstration into stage segments that enable targeted memory supervision for precise action generation. To identify meaningful task stage boundaries, we first select a well-performed demonstration as reference and extract its key poses that mark salient transitions such as grasps or directional changes from the end-effector's states $\mathbf{S}^{\text{ref}} = [\mathbf{s}_0^{\text{ref}}, \ldots, \mathbf{s}_n^{\text{ref}}]$ using the Ramer–Douglas–Peucker (RDP) algorithm [19]. The key poses are selected iteratively to preserve essential shape of the trajectory and the stage segmentation $\{\mathcal{S}_1, \ldots, \mathcal{S}_K\}$ is defined by placing each segment boundary at the midpoint between two consecutive key poses, enabling memory to guide the execution throughout the entire key pose. Furthermore, to ensure stage consistency across demonstrations of the same task, we employ the Dynamic Time Warping (DTW) algorithm [20], a technique that non-linearly aligns two sequences by minimizing their cumulative distance. Specifically, we align each trajectory $\mathbf{S}^i$ from the training demonstrations to the RDP-segmented reference trajectory $\mathbf{S}^{\text{ref}}$ by computing the optimal warping path that best matches their temporal progression. This process adjusts for variations in execution speed and duration while preserving the semantic structure of the task. As

a result, we obtain K stage-aligned segments $\mathcal{S}_1, \ldots, \mathcal{S}_K$ across all demonstrations, ensuring that the k-th segment in each demonstration consistently corresponds to the same task stage.

Stage-specific prompt tuning. Given the segmented and aligned demonstrations, we assign a soft prompt vector for each task stage that encodes stage-specific memory to guide the model in executing the corresponding key pose. At each timestep t, the observation $\mathbf{o}_t$ is processed by the image and language encoders to generate a base prompt with m base token embeddings $\mathcal{P}_{\text{base}} = [\mathbf{p}_1, \ldots, \mathbf{p}_m]$ from the immediate visual and textual inputs. To incorporate stage-specific memory into large VLA models, we augment the base prompt with a set of learnable soft tokens for each stage. These tokens are optimized via prompt tuning to encode abstract memory derived from training demonstrations, allowing the model to condition its behavior on the current task stage. Specifically, for a given stage $\mathcal{S}_k$, we define a learnable vector sequence $\mathcal{V}_k = [\mathbf{v}_1^k, \ldots, \mathbf{v}_m^k]$ as soft prompt. The final stage-specific memory prompt $\mathcal{P}_k$ input to the VLA model is computed via element-wise addition:

$$[\mathcal{P}_k]_j = [\mathcal{P}_{\text{base}}]_j + [\mathcal{V}_k]_j, \quad \forall j = 1, \ldots, m. \tag{2}$$

Since $\mathcal{P}_{\text{base}}$ is conditioned on $\mathbf{o}_t$, which varies within a stage, this additive formulation enables efficient integration of contextual memory without compromising the model's base capability and real-time adaptability. To encode the stage-specific memory, we optimize $\mathcal{V}_k$ by aligning the model's predicted action tokens with expert actions using the flow matching loss:

$$\mathcal{V}_k^* = \arg\min_{\mathcal{V}_k} \mathbb{E}_{p(\mathbf{A}_t|\mathbf{o}_t),\, q(\mathbf{A}_t^\tau|\mathbf{A}_t)} \left\| \mathbf{f}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \mathcal{V}_k) - \mathbf{u}(\mathbf{A}_t^\tau \mid \mathbf{A}_t) \right\|^2, \quad t \in \mathcal{S}_k. \tag{3}$$

This process effectively encodes the demonstration memory of a stage into the prompt embeddings to augment action generation. Finally, by tuning the vector sequences for all stages of a task, we construct the task memory prompts as $\{\mathcal{V}_k\}_{k=1}^K$. Repeating this process for all tasks yields a memory library, a repository of stage-specific prompts that encapsulates contextual cues as retrievable memory. We also add in the demonstration trajectories $\{\mathbf{S}^i\}_{i=1}^N$ and action sequences $\{\mathbf{A}^i\}_{i=1}^N$ in this library, where N is the total number of training demonstrations. This library is subsequently used to augment action generation in the following component.

C. Memory-Augmented Action Generation

Memory retrieval. To effectively retrieve relevant memory during task execution, the system must relate the robot's ongoing behavior with historical demonstrations. We address this by comparing the similarity between the robot's ongoing trajectory and the training demonstrations. Specifically, we define a fixed window size W and denote the last W steps of the robot's ongoing states up to time t as $\mathbf{S}^{\text{cur}} = [\mathbf{s}_{t-W+1}^{\text{cur}}, \ldots, \mathbf{s}_t^{\text{cur}}]$. For each demonstration trajectory $\mathbf{S}^i$, we perform a sliding window search to compute the similarity between the recent segment of $\mathbf{S}^{\text{cur}}$ and candidate segments from $\mathbf{S}^i$. For each candidate index j in $\mathbf{S}^i$, we compute the ℓ_2 distance as cost $\mathcal{C}$ between the current trajectory window and the corresponding reference window:

$$\mathcal{C}_{i,j} = \left\| \mathbf{S}^{\text{cur}} - \mathbf{S}^i[j - W + 1 : j] \right\|_2. \tag{4}$$

However, since long-horizon tasks often yield lengthy trajectories, the sliding window search can become computationally expensive, and similar trajectory windows may appear across different task stages. To address this, we adopt a hierarchical strategy by restricting candidate indices j to those whose associated stage label k_j^i and the most recently identified stage label k^{cur} satisfy $|k_j^i - k^{\text{cur}}| \leq 1$, thereby ensuring retrieval is limited to neighboring task stages. Then, we select the reference index j^* and trajectory i^* minimizing the distance:

$$i^*, j^* = \arg\min_{i,j} \mathcal{C}_{i,j}. \tag{5}$$

The task stage associated with (i^*, j^*) is then used to update the current task stage k^{cur} and retrieve the corresponding $\mathcal{V}_{k^{\text{cur}}}$ from the memory prompt library. This retrieved prompt reflects the memory of expert behavior for the most similar historical situation encountered from demonstrations.

Memory-aware prompt ensembling. While retrieving memory prompts provides valuable memory from past demonstrations tailored to the current stage of execution, the base prompt, trained over the entire task, provides broader task-level generalization. However, each alone is insufficient. The retrieved memory is vulnerable to retrieval inaccuracies, ambiguous execution stages, and stage misalignment of training demonstrations. Conversely, the base prompt, derived solely from current observations and task instructions, is not affected by such errors but lacks historical grounding and long-horizon memory. To reconcile these complementary strengths, we introduce a memory-aware prompt ensembling mechanism that dynamically combines the stage specificity of memory prompts with the task generalization capability of base prompts. At each timestep t, the frozen VLA model produces two action predictions: $\mathbf{A}_t^{\text{base}}$ using the base prompt $\mathcal{P}_{\text{base}}$, and $\mathbf{A}_t^{\text{mem}}$ using the retrieved stage-specific memory prompt $\mathcal{P}_{k^*}$. In parallel, we retrieve reference actions $\mathbf{A}_j^i = [\mathbf{a}_j^i, \ldots, \mathbf{a}_{j+H-1}^i]$ from demonstration $i = i^*$, starting at index $j = j^*$. This action sequence encapsulates valuable action priors about the best-matching demonstration's future actions under a similar situation. However, due to differences between the ongoing task and the historical demonstration, directly executing $\mathbf{A}_j^i$ is not feasible. Instead, we use it as an action prior to guide the dynamic weighting between $\mathbf{A}_t^{\text{base}}$ and $\mathbf{A}_t^{\text{mem}}$. Specifically, we compare these predictions to the retrieved demonstration action $\mathbf{A}_j^i$ and quantify a dynamic weighting coefficient α_t computed via softmax normalization:

$$\alpha_t = \frac{\exp(-\left\|\mathbf{A}_t^{\text{mem}} - \mathbf{A}_j^i\right\|^2)}{\exp(-\left\|\mathbf{A}_t^{\text{mem}} - \mathbf{A}_j^i\right\|^2) + \exp(-\left\|\mathbf{A}_t^{\text{base}} - \mathbf{A}_j^i\right\|^2)}. \tag{6}$$

TABLE I

PERFORMANCE COMPARISON ON LIBERO-LONG SIMULATION BENCHMARK. FOR EACH METHOD, THE FIRST ROW REPORTS SUCCESS RATES, AND THE SECOND ROW REPORTS STANDARD DEVIATIONS.

Method	Task1	Task2	Task3	Task4	Task5	Task6	Task7	Task8	Task9	Task10	Avg
OpenVLA	75.3% ± 8.1%	30.7% ± 4.2%	64.0% ± 3.5%	62.0% ± 2.0%	68.0% ± 3.5%	41.3% ± 3.1%	52.0% ± 1.2%	50.0% ± 0.0%	42.7% ± 6.1%	54.0% ± 5.3%	54.0% ± 0.4%
π_0	80.0% ± 4.0%	38.0% ± 3.5%	88.0% ± 4.0%	88.0% ± 4.0%	90.7% ± 1.2%	85.3% ± 2.3%	81.3% ± 3.1%	68.7% ± 12.2%	66.7% ± 11.4%	77.3% ± 4.6%	76.4% ± 2.3%
MAP-VLA	**92.0%** ± 2.0%	**42.7%** ± 2.3%	**96.0%** ± 2.0%	**90.7%** ± 1.2%	**93.3%** ± 1.2%	**93.3%** ± 3.1%	**90.7%** ± 4.2%	**75.3%** ± 1.2%	**69.3%** ± 6.4%	**90.7%** ± 6.1%	**83.4%** ± 0.7%

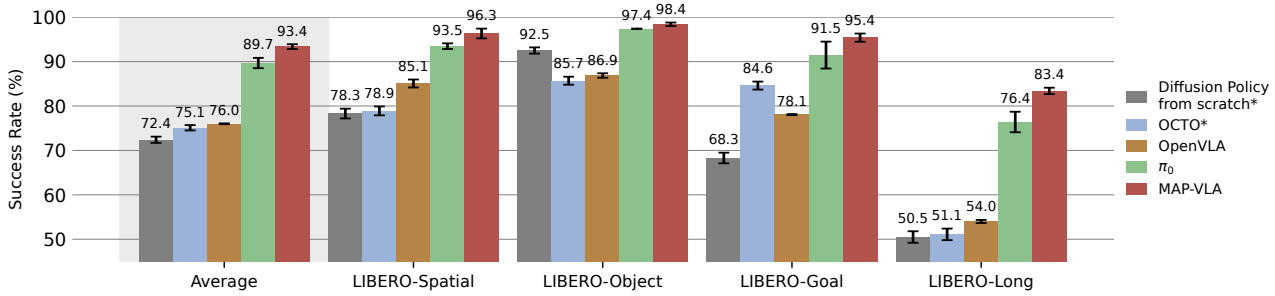

Fig. 3. Performance comparison on all LIBERO task suites, "*" denotes results reported by OpenVLA [3].

Using the computed α_t, the final memory-augmented action for execution is given by:

$$\mathbf{A}_t^{\text{MemAug}} = \alpha_t \mathbf{A}_t^{\text{mem}} + (1 - \alpha_t)\mathbf{A}_t^{\text{base}}. \tag{7}$$

This prompt ensemble mechanism encourages the policy to favor the action prediction that is closer to the retrieved expert action, effectively leveraging the future-action priors from demonstrations without sacrificing the VLA model's adaptability to the current scene. By dynamically balancing the task-level generalization of the base prompt with the stage-specificity of the retrieved prompt, the model maintains robustness to retrieval inaccuracies, improves tolerance to stage boundary ambiguity, and preserves adaptability to dynamic test-time scenarios, which results in more stable and accurate behavior during task execution.

In summary, MAP-VLA begins with stage segmentation and tuning stage-specific memory prompts for offline memory construction. The online execution loop then proceeds with (a) observing, (b) retrieving memory, (c) executing dual forward passes, and (d) prompt ensembling via dynamic weighting at each timestep.

IV. EXPERIMENTS

A. Implementation Details

The proposed MAP-VLA is based on the π_0 model [4]. We first follow [4] to fine-tune the π_0 model on the fine-tuning dataset using LoRA [21] on a server with 6 NVIDIA RTX 6000 Ada GPUs, and then freeze the model weights and apply the same fine-tuning configuration for prompt tuning. For simulation experiments, we use the LIBERO benchmark [22] and adopt the training data and evaluation settings of OpenVLA [3], where the success rate is the average over 3 random seeds x 50 rollouts for each task. For real-world experiments, MAP-VLA is deployed on a 6-DoF Galaxea A1 robotic arm shown in Fig. 4 to perform 20 rollouts for each task. All real-world computations are conducted on a system with an NVIDIA RTX 4090 GPU.

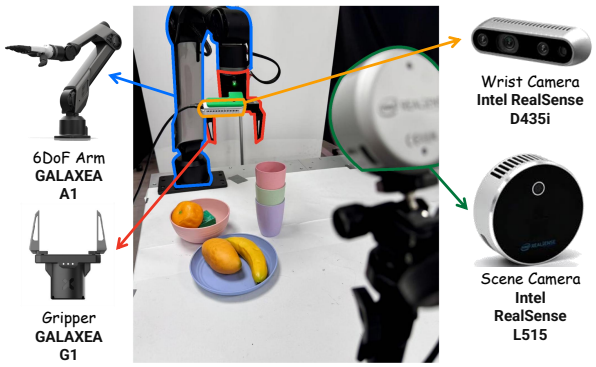

Fig. 4. Real-world environment setup.

B. Comparison on Long-Horizon Tasks

Comparison on LIBERO simulation. We first compare MAP-VLA to baseline VLA models, OpenVLA [3] and π_0 [4], on the challenging long-horizon manipulation tasks suite LIBERO-Long (also referred to as LIBERO-10) from the LIBERO simulation benchmark. The set includes: (Task1) *pick up the book and place it in the back compartment of the caddy*, (Task2) *put both moka pots on the stove*, (Task3) *put both the alphabet soup and the cream cheese box in the basket*, (Task4) *put both the alphabet soup and*

TABLE II

PERFORMANCE COMPARISON OF LONG-HORIZON TASK SUCCESS RATES ON REAL ROBOT EVALUATIONS.

Task	Partial Success				Complete Success			
	π_0	MAP-VLA	Abs. Gain	Rel. Gain	π_0	MAP-VLA	Abs. Gain	Rel. Gain
Task1	55%	**70%**	15%	27.3%	35%	**50%**	15%	42.9%
Task2	55%	**75%**	20%	36.4%	25%	**55%**	30%	120.0%
Task3	50%	**60%**	10%	20.0%	10%	**40%**	30%	300.0%
Avg	53.3%	**68.3%**	15.0%	28.1%	23.3%	**48.3%**	25.0%	107.1%

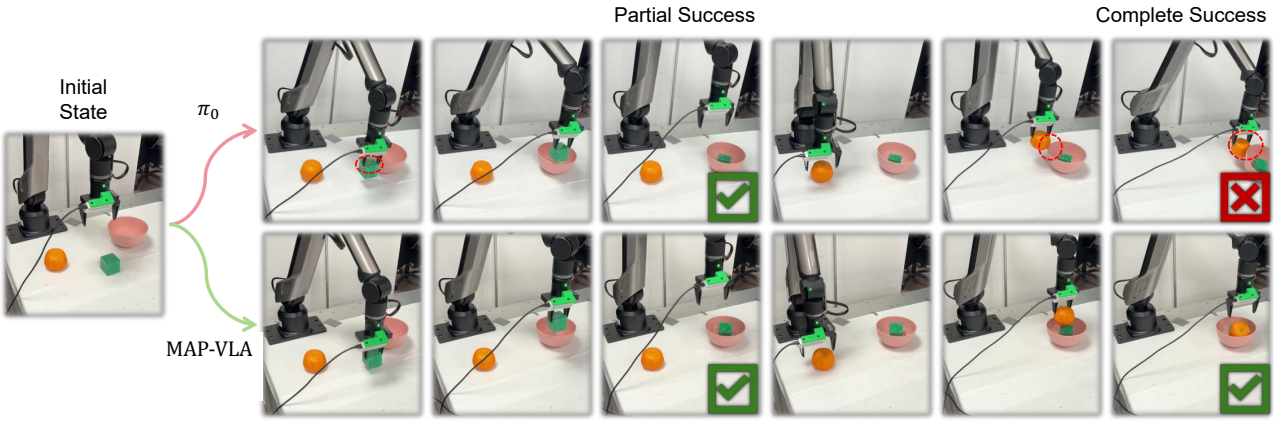

Fig. 5. Visualization and comparison of Task2: *place the green cube and orange into the bowl*.

the tomato sauce in the basket, (Task5) *put both the cream cheese box and the butter in the basket*, (Task6) *put the black bowl in the bottom drawer of the cabinet and close it*, (Task7) *put the white mug on the left plate and put the yellow and white mug on the right plate*, (Task8) *put the white mug on the plate and put the chocolate pudding to the right of the plate*, (Task9) *put the yellow and white mug in the microwave and close it*, and (Task10) *turn on the stove and put the moka pot on it*. As shown in Table I, MAP-VLA consistently outperforms all baselines across every individual task in this benchmark. On average, MAP-VLA achieves an 83.4% success rate, whereas the baseline OpenVLA and π_0 achieve 54.0% and 76.4%, respectively. This corresponds to a 7.0% absolute and 9.2% relative gains over π_0, the strongest memoryless baseline. These results underscore the advantage of equipping VLA models with retrievable, stage-specific memory derived from expert demonstrations, particularly in scenarios where long-horizon task decomposition and contextual grounding are critical for success. We also note that MAP-VLA's trial outcomes are more consistent, with a lower standard deviation in success rate (0.7%) across runs than π_0 (2.3%). This reduced variability suggests improved robustness and reliability, as a result of encoding additional contextual memory into the prompt and dynamic prompt ensembling as we discuss in Section IV-E. We also evaluate MAP-VLA on the full LIBERO benchmark, the results presented in Fig. 3 demonstrate that our method consistently outperforms all baselines across all task suites, highlighting its superior capability beyond long-horizon settings.

Comparison on real robot evaluations. To validate the real-world effectiveness of MAP-VLA, we conduct evaluations on a physical robotic platform and compare its performance with the strongest baseline, π_0, across three long-horizon tasks. Each task comprises two sequential subtasks. The evaluated tasks include: (Task1) *place the banana and mango into the plate*, (Task2) *place the green cube and orange into the bowl*, and (Task3) *stack the green cup and pink cup on the purple cup*. We report both "Partial Success" (first sub-task completed) and "Complete Success" (both sub-tasks completed). As summarized in Table II, MAP-VLA again outperforms the baseline policy. Averaged over the three tasks, MAP-VLA's partial success and complete success rates are 68.3% and 48.3%, versus 53.3% and 23.3% for the baseline, showing a substantial improvement in overall task completion. Notably, the performance gap is more pronounced in complete success in terms of both absolute gains and relative gains, suggesting that memory augmentation plays a particularly critical role in sustaining correct behavior across extended action sequences. Overall, the comparisons confirm that MAP-VLA sets a new state-of-the-art for long-horizon task execution in both simulation and real-robot settings, with significantly higher success rates than prior methods.

Case study. We showcase a specific real-world case to further demonstrate the key improvements of our method. We focus on Task2, where the edges and relative sizes of two objects pose a significant challenge for precise grasping and placement. Such configurations require the policy to distinguish fine-grained spatial cues and execute careful actions. The comparison is visualized in Fig. 5. Effective

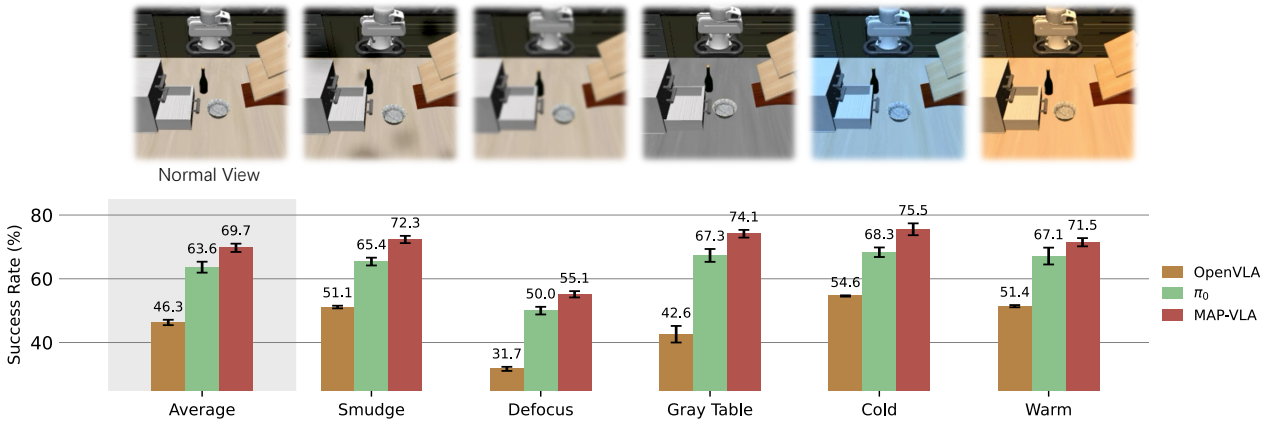

Fig. 6. Performance comparison with visual variations on LIBERO-Long.

TABLE III
ABLATION STUDY ON LIBERO-LONG.

Metric	Base VLA	Universal Prompt	Task Prompt	Stage Prompt	MAP-VLA
Success Rate (SR)	76.4%	76.9%	79.3%	81.4%	**83.4%**
Standard Deviation (Std)	±2.3%	±2.4%	±0.8%	±2.3%	±0.7%

long-horizon robot manipulation often demands fine-grained memory to maintain a coherent trajectory across multiple stages. However, the memoryless baseline policy π_0 exhibits inconsistent and ambiguous object alignment behavior, especially during critical pick-and-place phases (as circled in the figure), often leading to task failure. In contrast, our MAP-VLA framework demonstrates memory-augmented robustness in such settings. By retrieving and incorporating temporally relevant prompts, it enables more stable, context-aware action generation that better adheres to task constraints and results in successful execution under spatial variations.

C. Comparison with Visual Variations

To assess robustness under real-world visual variations, we evaluate MAP-VLA on LIBERO-Long tasks subjected to various challenging visual conditions. As illustrated in Fig. 6, we introduce several types of visual perturbations to the robot's observations during test time: (i) smudge – simulating dirty camera lens; (ii) defocus – simulating an out-of-focus camera; (iii) gray table – a shift in table color from the training demonstrations; (iv) cold – a bluish, low-temperature illumination; and (v) warm – a yellowish, high-temperature lighting condition. Despite these visual shifts, MAP-VLA consistently retains a significantly higher success rate compared to the baseline π_0 policy. For instance, under the smudge condition, MAP-VLA maintains 72.3% success while π_0 drops to 65.4%, marking the highest relative gain of 10.6%. In defocused settings, where fine visual features are blurred, MAP-VLA still outperforms π_0 (55.1% versus 50.0%), aided by memory-based inference from prior demonstrations. MAP-VLA also performs well under table color change, which can be a practical use case, with 74.1% versus 67.3%. Overall, MAP-VLA achieves an average relative gain of 9.6%, slightly above the 9.2% relative gain without visual variations. Collectively, these findings underscore that memory-augmented prompting remains robust under perceptual variations, suggesting strong generalization capabilities across diverse visual domains.

TABLE IV
COMPARISON UNDER 10-SHOT AND 20-SHOT ON LIBERO-LONG.

Metric	10-shot		20-shot	
	π_0	MAP-VLA	π_0	MAP-VLA
SR	53.6%	**55.8%**	72.1%	**75.9%**
Std	±1.1%	±0.9%	±2.0%	±0.8%

D. Comparison under Few-Shot Setting

To further evaluate the adaptability of our method, we compare it with baseline approaches under few-shot learning scenarios. Specifically, the training data contains only 10 or 20 demonstrations per task. Table IV summarizes the performance of MAP-VLA and π_0 on the LIBERO-Long benchmark. MAP-VLA achieves average success rates of 55.8% and 75.9% for the 10-shot and 20-shot settings, which are consistently higher than those of the baseline π_0 at 53.6% and 72.1%. Moreover, MAP-VLA exhibits lower standard deviation (±0.9% and ±0.8%) compared to π_0 (±1.1% and ±2.0%), indicating improved robustness even with limited training data.

E. Ablation Study

We conduct an ablation study on the LIBERO-Long benchmark to quantify the impact of each component in MAP-VLA. Table III summarizes the average success rates

and standard deviations across five progressively enhanced model variants: (a) Base VLA (π_0): The original π_0 model without any prompt tuning or memory augmentation. (b) Universal Prompt: A single soft prompt shared and optimized across all tasks. (c) Task Prompt: A distinct soft prompt per task, applied over the entire trajectory. (d) Stage Prompt: A collection of stage-specific memory prompts, providing fine-grained guidance through retrieval. (e) MAP-VLA: Our complete framework combining stage-specific memory prompts with memory-aware prompt ensembling.

Impact of memory-aware prompt ensembling. The full MAP-VLA (e) variant adds memory-aware prompt ensembling, which dynamically balances the influence of the stage-specific prompt and the base prompt based on their agreement with retrieved demonstration actions. This integration yields the best overall performance, achieving an 83.4% success rate and the lowest observed standard deviation (0.7%), reflecting increased consistency and robustness. Crucially, this ensembling strategy enables the model to leverage the complementary strengths of both the task-general base prompt and the stage-specific memory prompts, guided by the retrieved demonstration as action priors. On the other hand, variant (d) also achieves a success rate of 81.4%, offering a more computationally efficient alternative by avoiding dual forward passes.

F. Retrieval Latency

In MAP-VLA, memory retrieval is limited to the current and neighboring stages of a task. Assuming task stages are of comparable length, the retrieval cost depends solely on the number of demonstrations for that task and is independent of the total number of tasks and their trajectory lengths. The resulting complexity is $\mathcal{O}(N)$, where N denotes the number of training demonstrations for the task. In LIBERO-Long, with an average of 37.8 demonstrations per task, the average retrieval time is only 21.6 ms.

V. CONCLUSIONS

We propose MAP-VLA, a memory-augmented framework that equips a pre-trained Vision-Language-Action (VLA) model with demonstration-derived memory prompts to enhance long-horizon robotic manipulation. By combining prompt tuning with retrieval augmentation, MAP-VLA allows a frozen VLA model to dynamically retrieve and utilize stage-specific memory from expert demonstrations and augment action generation. This flexible and lightweight design improves performance significantly across both simulated benchmark and real-world tasks, achieving higher success rates and increased robustness under visual variations. Our work on memory-augmented VLA contributes positively to the development of more robust and adaptable assistive robots, with potential applications in household support, eldercare, and industrial automation. Looking ahead, promising directions for future research include developing generalized yet informative memory prompts that can be reused flexibly across a broad spectrum of tasks, reducing the need for stage-specific memory construction.

REFERENCES

[1] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *Journal of machine learning research*, vol. 22, no. 30, pp. 1–82, 2021.

[2] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," in *Conference on Robot Learning*. PMLR, 2023, pp. 2165–2183.

[3] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi *et al.*, "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[4] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter *et al.*, "π_0: A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.

[5] H. Zhen, X. Qiu, P. Chen, J. Yang, X. Yan, Y. Du, Y. Hong, and C. Gan, "3d-vla: A 3d vision-language-action generative world model," *arXiv preprint arXiv:2403.09631*, 2024.

[6] Q. Zhao, Y. Lu, M. J. Kim, Z. Fu, Z. Zhang, Y. Wu, Z. Li, Q. Ma, S. Han, C. Finn *et al.*, "Cot-vla: Visual chain-of-thought reasoning for vision-language-action models," *arXiv preprint arXiv:2503.22020*, 2025.

[7] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6892–6903.

[8] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine, "Bridge data: Boosting generalization of robotic skills with cross-domain datasets," *arXiv preprint arXiv:2109.13396*, 2021.

[9] L. Fu, H. Huang, G. Datta, L. Y. Chen, W. C.-H. Panitch, F. Liu, H. Li, and K. Goldberg, "In-context imitation learning via next-token prediction," *arXiv preprint arXiv:2408.15980*, 2024.

[10] Y. Yin, Z. Wang, Y. Sharma, D. Niu, T. Darrell, and R. Herzig, "In-context learning enables robot action prediction in llms," *arXiv preprint arXiv:2410.12782*, 2024.

[11] V. Vosylius and E. Johns, "Instant policy: In-context imitation learning via graph diffusion," *arXiv preprint arXiv:2411.12633*, 2024.

[12] J. Y. Zhu, C. G. Cano, D. V. Bermudez, and M. Drozdzal, "Incoro: In-context learning for robotics control with feedback loops," *arXiv preprint arXiv:2402.05188*, 2024.

[13] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," *arXiv preprint arXiv:2104.08691*, 2021.

[14] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Learning to prompt for vision-language models," *International Journal of Computer Vision*, vol. 130, no. 9, pp. 2337–2348, 2022.

[15] H. Yao, R. Zhang, and C. Xu, "Tcp: Textual-based class-aware prompt tuning for visual-language model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 23 438–23 448.

[16] R. Li, Y. Chen, Z. Weng, Z. Lin, and Y.-P. Tan, "Class-specific prompt learning for vision–language models," *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

[17] M. Jia, L. Tang, B.-C. Chen, C. Cardie, S. Belongie, B. Hariharan, and S.-N. Lim, "Visual prompt tuning," in *European conference on computer vision*. Springer, 2022, pp. 709–727.

[18] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PmLR, 2021, pp. 8748–8763.

[19] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Computer graphics and image processing*, vol. 1, no. 3, pp. 244–256, 1972.

[20] M. Müller, "Dynamic time warping," *Information retrieval for music and motion*, pp. 69–84, 2007.

[21] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models." *ICLR*, vol. 1, no. 2, p. 3, 2022.

[22] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, "Libero: Benchmarking knowledge transfer for lifelong robot learning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 44 776–44 791, 2023.