arXiv:2608.14986v1 [cs.RO] 15 Aug 2026

GaussMemory: Task-Driven 3D Gaussian Scene Memory for Long-Horizon Robotic Manipulation

Zhiqiang HU, Shouren HUANG and Masatoshi ISHIKAWA

***Abstract*— Long-horizon robotic manipulation fundamentally relies on persistent spatial memory. However, existing 3D memory systems function merely as passive recorders: they store observations using fixed, hand-crafted rules, treating every scene element—whether a critical grasp target or an irrelevant background wall—with equal importance. In this paper, we propose a paradigm shift from passive storage to active, task-driven spatial memory. We argue that a robot's memory should not simply record what it sees, but actively learn *how* to remember—discovering which objects to track precisely, how aggressively to update them, and what to discard, all learned end-to-end without hand-designed rules. Crucially, this active paradigm is realized by unifying memory update and readout as two sides of the same cognitive process, enabling bidirectional flow where task needs shape update strategies and vice versa. To instantiate this vision, we introduce GaussMemory, which leverages 3D Gaussian Splatting as a persistent geometric substrate. On LIBERO, GaussMemory outperforms MemoryVLA on Goal and Long-10; on VLABench, it surpasses π_0-FAST by +5.2% (Track 1) and +6.0% (Track 6).**

I. Introduction

Vision-Language-Action (VLA) models such as OpenVLA [9] and π_0 [2] have emerged as a powerful paradigm for robotic manipulation, leveraging pretrained vision-language knowledge to map visual observations and language instructions directly to actions. However, memoryless VLA systems are mainly limited to tasks in which the next action can be inferred from the current sensory observation. As Fig. 1 illustrates, when an object is stored in one of two visually identical drawers and subsequently becomes occluded, the current RGB observation is insufficient to identify the correct drawer after an intervening sub-task. Persistent spatial memory is therefore required to retain the object's identity and last observed 3D position across time, enabling reliable retrieval in long-horizon manipulation.

Recent efforts have begun to equip VLA systems with temporal memory, but the solutions remain limited. MemoryVLA [18] maintains a 2D perceptual-cognitive memory bank achieving strong results, yet its memory is fundamentally 2D and cannot answer spatial queries such as "where exactly is the plate relative to the glass?" MEM [20] introduces multi-scale embodied memory combining semantic, episodic, and procedural stores, but likewise operates in 2D token space without explicit 3D geometry. Language-based approaches such as SayPlan [17] record text descriptions of past states but sacrifice geometric precision entirely. Even recent 3D-structured systems such as ConceptGraphs [4] that maintain scene graphs with explicit geometry still rely on hand-crafted update rules—thresholded matching, fixed-rate averaging, hard pruning—entirely decoupled from task performance. Across all these categories, the memory systems share a common limitation: they function as *passive recorders* that treat every scene element with equal importance. Crucially, this stems from a deeper issue: memory update and readout are treated as independent operations, with no communication between them. We propose to move beyond this passive paradigm by shifting to *active, task-driven* 3D spatial memory within the VLA framework (Fig. 2). The memory should learn not just *what* to remember, but *how* to remember—which objects to track with high precision, which observations to trust, when to update aggressively and when to preserve—all guided by one criterion: *does this way of remembering help the robot complete its task?* The key insight is that memory update and readout are not separate processes but two sides of the same cognitive computation: how you *use* memory (readout) should directly shape how you *store* it (update), and vice versa. For instance, if the next sub-task requires placing silverware next to the plate, the readout queries will attend strongly to the plate's position; this attention signal should flow back to bias the update toward tracking the plate more precisely. Conversely, if an object like a cup was just moved, the update event should inform the readout to trust its new position more. This bidirectional coupling—inspired by cognitive science phenomena like the Testing Effect, where retrieval strengthens memory encoding—is absent in existing

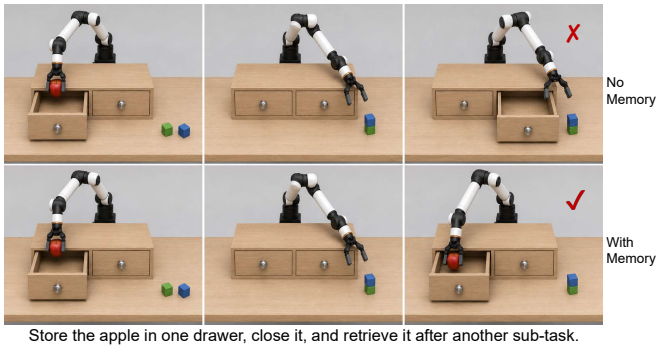

Fig. 1. **With vs. without persistent memory.** Why persistent 3D memory is necessary. The robot stores an apple in one of two visually identical drawers and completes an intervening sub-task. Once both drawers are closed, the current RGB observation no longer reveals the apple's location. A memoryless policy cannot disambiguate the correct drawer.

The authors are with the Research Institute for Science & Technology, Tokyo University of Science {zhiqiang.hu, huang, ishikawa}@ishikawa-vision.org

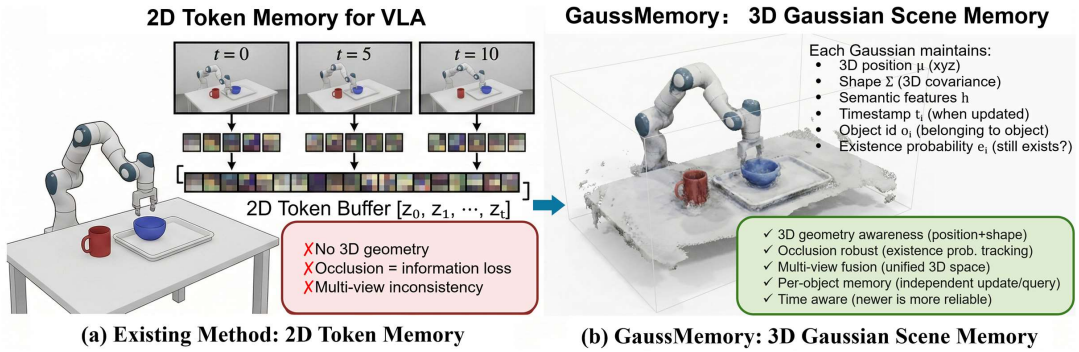

Fig. 2. **2D token memory vs. 3D Gaussian scene memory. (a)** Existing VLA methods buffer 2D image tokens that lack 3D geometry, lose information under occlusion, and suffer from multi-view inconsistency. **(b)** GaussMemory maintains a persistent 3D Gaussian Scene Memory where each Gaussian carries explicit position, shape, semantic features, timestamps, object identity, and existence probability.

decoupled systems but enables the task loss to genuinely sculpt memory behavior.

3D Gaussian Splatting (3DGS) [8] is a particularly suitable substrate for realizing active memory within VLA, thanks to three properties: *(1) Natural incrementality*: individual Gaussians can be added, removed, or repositioned without disrupting the rest of the scene. *(2) Explicit geometry*: each Gaussian carries position $\boldsymbol{\mu}$, covariance $\boldsymbol{\Sigma}$, and opacity α, providing the precise spatial information that manipulation planning requires. *(3) Tokenizability*: Gaussian features can be encoded into compact tokens and injected into an LLM backbone alongside language and 2D visual tokens, integrating naturally into the VLA architecture. Yet no existing system takes the step from passive Gaussian storage to active, task-driven Gaussian memory within VLA—where the update mechanism is end-to-end differentiable with respect to the action prediction loss. We realize this unified vision through **Unified Memory Attention (UMA)**, where observation tokens and readout queries are concatenated and processed in shared cross-attention layers. This design enables bidirectional information flow—the readout is aware of what just changed, and the update is aware of what the task cares about—so that the task's information need shapes memory behavior not only through gradients during training but also through forward attention flow at inference time.

We present GaussMemory with three contributions:

Active 3D Gaussian Memory Paradigm: We propose that robot spatial memory should shift from passive recording to active, task-driven learning, instantiated using 3D Gaussian Splatting as a persistent geometric substrate with temporal metadata and an object-centric scene graph. Empirically, the learned position-update coefficient exhibits a bimodal pattern, with an average value of approximately 0.74 for manipulated objects and 0.06 for background Gaussians.

Unified Memory Attention (UMA): A single cross-attention module that *simultaneously* reads memory for action prediction and writes updates back, making the entire memory lifecycle end-to-end differentiable. UMA enables bidirectional information flow: task needs shape memory updates, and memory state informs action readout.

Persistent 3D Gaussian Memory Integration for VLA: To our knowledge, GaussMemory is the first to integrate 3D Gaussian Splatting as a *temporal scene memory* within a VLA framework, where persistently updated memory tokens derived from 3D Gaussians via UMA are consumed by an LLM backbone for action prediction. Unlike prior works that encode static Gaussian scenes into tokens for language-based QA [5], our Gaussian memory is continuously updated across sub-tasks and jointly optimized with robot action loss.

II. Related Work

A. Memory for Long-Horizon Manipulation

Long-horizon robotic tasks require maintaining information across many interaction steps. Existing memory approaches fall into three categories:

Language-based memory: SayPlan [17] uses a language-annotated 3D scene graph updated via text descriptions. While compact and LLM-friendly, these approaches lose geometric precision—"the cup is on the counter" does not encode *where* on the counter or the cup's spatial relationship to nearby objects.

Image-based memory: MemoryVLA [18] maintains a 2D perceptual-cognitive memory bank with temporal positional encoding, achieving strong results on long-horizon tasks. However, its memory is fundamentally 2D token-level—it cannot answer spatial queries such as "where is the cup relative to the plate?" that require 3D geometric reasoning.

3D-structured memory: ConceptGraphs [4] builds object-centric 3D scene graphs from multi-view observations. All these systems separate memory update from memory readout into independent non-differentiable stages.

GaussMemory differs from all the above by shifting from passive to *active* memory: the update mechanism itself is learned from task outcomes, and 3D Gaussians provide the geometric precision that text and image memories lack.

B. 3D Gaussian Splatting for Robotics

3DGS [8] has expanded into robotic perception. ManiGaussian [13] uses dynamic Gaussians as a forward world model, whereas GaussMemory uses them as a persistent scene memory for retrospective state tracking and future

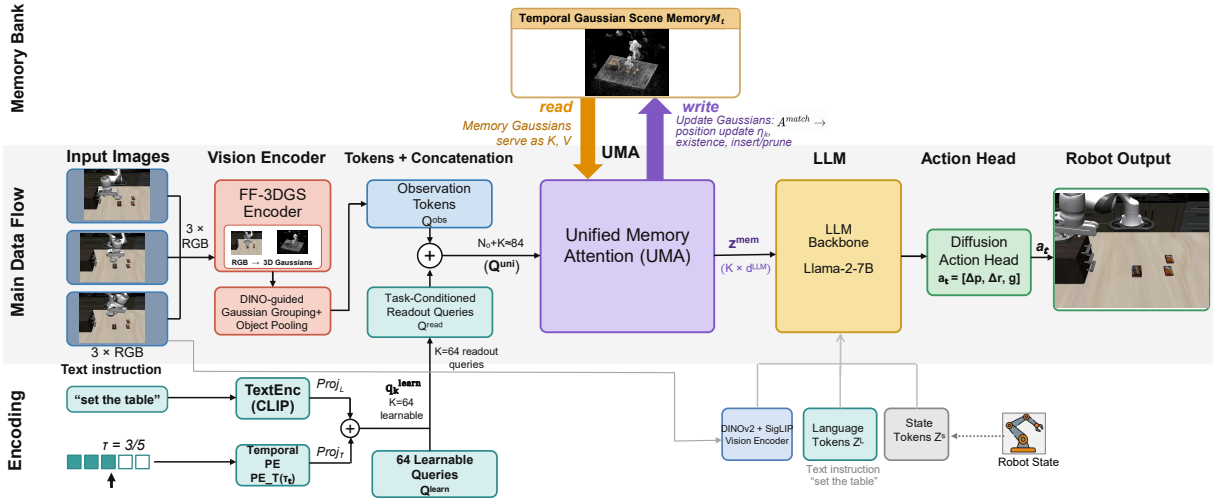

Fig. 3. **GaussMemory architecture.** Multi-view RGB images are encoded by FF-3DGS into observation tokens $\mathbf{Q}^{\text{obs}}$, which are concatenated with task-conditioned readout queries $\mathbf{Q}^{\text{read}}$ and jointly processed by **Unified Memory Attention (UMA)**. UMA cross-attends to the 3D Gaussian Scene Memory (K/V up to 32K persistent Gaussians), producing two parallel outputs: the *output stream* $\mathbf{Z}^{\text{mem}}$ provides 3D spatial context to the LLM backbone for action prediction via the Diffusion Action Head, while the *update memory stream* writes back updated Gaussians via learned gates η_k, matching, insertion, and pruning.

action prediction. GaussianVLM [5] encodes static Gaussian scenes for language-based reasoning and produces text outputs. In contrast, GaussMemory maintains a temporally evolving Gaussian memory whose tokens directly condition robot action prediction.

III. Method

A. Problem Formulation

We consider long-horizon manipulation tasks specified by a high-level language instruction $\mathcal{L}$ that decomposes into a sequence of T sub-tasks $\{\tau_1, \ldots, \tau_T\}$. At each time step t, the robot receives multi-view RGB images $\{I_v^t\}_{v=1}^V$ and its proprioceptive state $\boldsymbol{s}_t$. The goal is to predict actions $\boldsymbol{a}_t$ that sequentially complete all sub-tasks while maintaining persistent memory of scene changes.

B. System Overview

As shown in Fig. 3, GaussMemory consists of three components: a *Temporal Gaussian Scene Memory* $\mathcal{M}_t$ (persistent 3D Gaussians), *Unified Memory Attention (UMA)* (simultaneous memory read/write via shared cross-attention), and a Prismatic VLM backbone (Llama-2-7B, shared with OpenVLA/CogACT/MemoryVLA) with a diffusion action head.

C. Temporal Gaussian Scene Memory

The core data structure is the Temporal Gaussian Scene Memory $\mathcal{M}_t$, a set of augmented Gaussian primitives maintained persistently across the task horizon.

a) Augmented Gaussian Primitives: Each Gaussian carries standard 3DGS attributes plus temporal metadata:

$$G_i = (\underbrace{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \alpha_i, \boldsymbol{c}_i, \boldsymbol{h}_i}_{\text{standard 3DGS}}, \underbrace{t_i^c, t_i^m, \tau_i, o_i, e_i}_{\text{temporal metadata}}) \tag{1}$$

where $\boldsymbol{\mu}_i \in \mathbb{R}^3$ is position, $\boldsymbol{\Sigma}_i$ the covariance, α_i opacity, $\boldsymbol{c}_i$ color, $\boldsymbol{h}_i$ spherical harmonic features, t_i^c/t_i^m creation/modification timestamps, τ_i sub-task index, and $o_i \in \{1, \ldots, O\}$ object ID, and existence probability $e_i \in (0, 1)$.

b) Temporal Feature Encoding: Each Gaussian is encoded as:

$$\begin{aligned}\boldsymbol{g}_i^{\text{mem}} = \text{MLP}_{\text{enc}}\big([\boldsymbol{\mu}_i; \text{vec}(\boldsymbol{\Sigma}_i); \alpha_i; \boldsymbol{h}_i]\big) \\ + \text{PE}_{3D}(\boldsymbol{\mu}_i) + \text{PE}_T^{\text{rel}}(t_i^m, t)\end{aligned} \tag{2}$$

where PE_{3D} is a 3D Fourier positional encoding [19], and $\text{PE}_T^{\text{rel}}(t_i^m, t) = \text{PE}_T(t - t_i^m)$ encodes the relative age of a memory entry.

$$\text{PE}_T(t) = \bigoplus_{n=1}^{N_T} \left[\sin(\omega_n^T t), \cos(\omega_n^T t)\right], \qquad \omega_n^T = \frac{2^{n-1}\pi}{T_{\max}}. \tag{3}$$

where ω_n^T denotes the fixed Fourier frequencies, $T_{\max}$ is the maximum episode length, and N_T is the number of frequency bands. We use relative encoding $\text{PE}_T^{\text{rel}}(t_i^m, t) = \text{PE}_T(t - t_i^m)$ for length generalization.

c) Object-Centric Organization: Object instance IDs are obtained via Gaussian Grouping [22] with DINO features, organizing Gaussians into per-object groups $\mathcal{G}_j = \{G_i : o_i = j\}$.

D. Unified Memory Attention (UMA)

UMA is the mechanism at the heart of GaussMemory. It solves two coupled problems—*what to update* and *what to retrieve*—in a single differentiable attention mechanism. By first letting observation tokens and task-conditioned readout queries exchange information via self-attention, and then jointly cross-attending to the 3D Gaussian memory, UMA produces both task-relevant spatial context for the LLM *and*

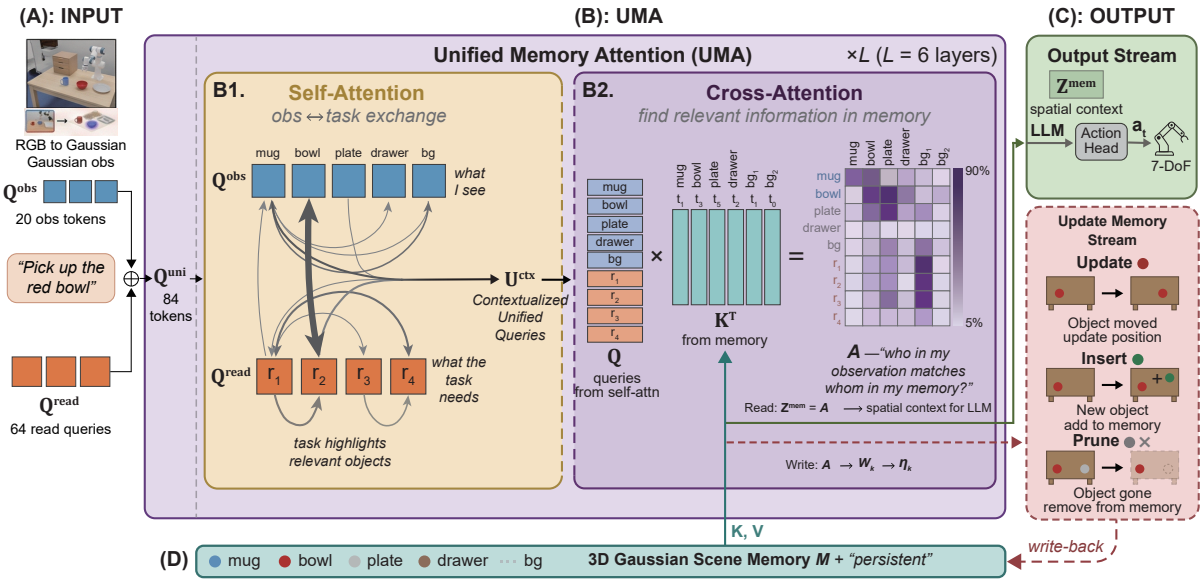

Fig. 4. **Inside Unified Memory Attention (UMA),** expanding the UMA block from Fig. 3. **(A)** Observation tokens and readout queries are concatenated. **(B1)** Self-attention exchanges task and perceptual information. **(B2)** Shared cross-attention queries Gaussian memory through keys and values, producing read-query features for $\mathbf{Z}^{\text{mem}}$ and observation-to-memory correspondences $\mathbf{A}^{\text{match}}$ for memory update. **(C)** The two streams support action prediction and memory update, insertion, and pruning. **(D)** Persistent 3D Gaussian Scene Memory.

soft correspondence signals for memory update. The data flow proceeds in four stages (Fig. 4):

a) Stage 1: Construct Unified Query: Observation tokens $\mathbf{Q}^{\text{obs}}$: FF-3DGS produces M live Gaussians from multi-view images, each encoded as $\boldsymbol{g}_i^{\text{live}} = \text{MLP}_{\text{enc}}([\boldsymbol{\mu}_i; \text{vec}(\boldsymbol{\Sigma}_i); \alpha_i; \boldsymbol{h}_i]) + \text{PE}_{3D}(\boldsymbol{\mu}_i)$, then aggregated to N_o object-level tokens:

$$\boldsymbol{q}_j^{\text{obs}} = \text{Pool}\big(\{\boldsymbol{g}_i^{\text{live}} : o_i = j\}\big) + \text{PE}_{3D}(\bar{\boldsymbol{\mu}}_j) \tag{4}$$

where $\bar{\boldsymbol{\mu}}_j = \frac{1}{|\mathcal{G}_j|}\sum_{i:o_i=j}\boldsymbol{\mu}_i$ is the object centroid.

Readout queries $\mathbf{Q}^{\text{read}}$: K learnable tokens conditioned on the language instruction $\mathcal{L}$ and current sub-task τ_t:

$$\boldsymbol{q}_k^{\text{read}} = \boldsymbol{q}_k^{\text{learn}} + \text{Proj}_L(\text{TextEnc}(\mathcal{L})) + \text{Proj}_\tau(\text{PE}_T(\tau_t)) \tag{5}$$

where $\boldsymbol{q}_k^{\text{learn}} \in \mathbb{R}^d$ are randomly initialized learnable embeddings, TextEnc is CLIP text encoder, and Proj_L, Proj_τ are linear projections.

The unified query concatenates both groups: $\mathbf{Q}^{\text{uni}} = [\mathbf{Q}^{\text{obs}}; \mathbf{Q}^{\text{read}}] \in \mathbb{R}^{(N_o+K)\times d}$.

b) Stage 2: Self-Attention — Observation-Task Exchange.: Self-attention enables bidirectional information flow within the unified query *before* accessing memory. Readout queries attend to observation tokens, "highlighting" task-relevant objects (instruction → observation). Conversely, observation tokens ground readout queries in the current scene state (observation → readout). The output is the **Contextualized Unified Query** $\mathbf{U}^{\text{ctx}}$ (Fig. 4), in which observation tokens have become *task-aware* and readout queries have become *perceptually grounded*—ready for targeted memory access.

c) Stage 3: Cross-Attention Against Memory.: $\mathbf{U}^{\text{ctx}}$ now queries the 3D Gaussian memory. Keys and values come from stored memory Gaussians $\mathbf{G}^{\text{mem}} \in \mathbb{R}^{N_m\times d}$ (N_m: number of memory Gaussians). Each of the L UMA layers applies self-attention, cross-attention, then FFN, each with LayerNorm (LN) and residual connections:

$$\hat{\mathbf{Q}}^{(l)} = \text{LN}\big(\text{SelfAttn}(\mathbf{Q}^{(l-1)}) + \mathbf{Q}^{(l-1)}\big) \tag{6}$$

$$\bar{\mathbf{Q}}^{(l)} = \text{LN}\big(\text{TimeAttn}(\hat{\mathbf{Q}}^{(l)}, \mathbf{G}^{\text{mem}}) + \hat{\mathbf{Q}}^{(l)}\big) \tag{7}$$

$$\mathbf{Q}^{(l)} = \text{LN}\big(\text{FFN}(\bar{\mathbf{Q}}^{(l)}) + \bar{\mathbf{Q}}^{(l)}\big) \tag{8}$$

where $\hat{\mathbf{Q}}^{(l)}$ corresponds to $\mathbf{U}^{\text{ctx}}$ in Fig. 4—the contextualized queries after self-attention exchange. The cross-attention incorporates temporal bias: $\mathbf{A} = \text{softmax}(\mathbf{Q}\mathbf{W}_Q(\mathbf{G}^{\text{mem}}\mathbf{W}_K)^\top/\sqrt{d_k} + \mathbf{B}_T)$ and $\text{TimeAttn}(\mathbf{Q}, \mathbf{G}^{\text{mem}}) = \mathbf{A}\,\mathbf{G}^{\text{mem}}\mathbf{W}_V$, where $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d\times d_k}$ ($d_k = d/H$, H heads), $\mathbf{B}_T$ modulates by temporal recency. $B_T \in \mathbb{R}^{(N_o+K)\times N_m}$ is a learned relative-time bias computed from $\text{PE}_T^{\text{rel}}(t_i^m, t)$ and shared across attention heads.

d) Stage 4: Read and Write Streams.: After L layers, the output splits: $\mathbf{Q}^{(L)} = [\mathbf{Q}^{\text{obs},(L)}; \mathbf{Q}^{\text{read},(L)}]$. The *same* cross-attention that reads from memory also produces write signals, eliminating separate matching (e.g., Sinkhorn or Hungarian).

Read stream → **LLM input**: $\mathbf{Z}^{\text{mem}} = \text{LinearProj}(\mathbf{Q}^{\text{read},(L)}) \in \mathbb{R}^{K\times d_{\text{LLM}}}$, where d_{LLM} is the LLM hidden dimension.

Write stream → **Memory update**: The observation rows of the final cross-attention produce a soft correspondence matrix $\mathbf{A}^{\text{match}} = \text{Attn}_{\text{cross}}^{(L)}[1{:}N_o, :] \in \mathbb{R}^{N_o\times N_m}$. This single matrix drives the entire memory lifecycle:

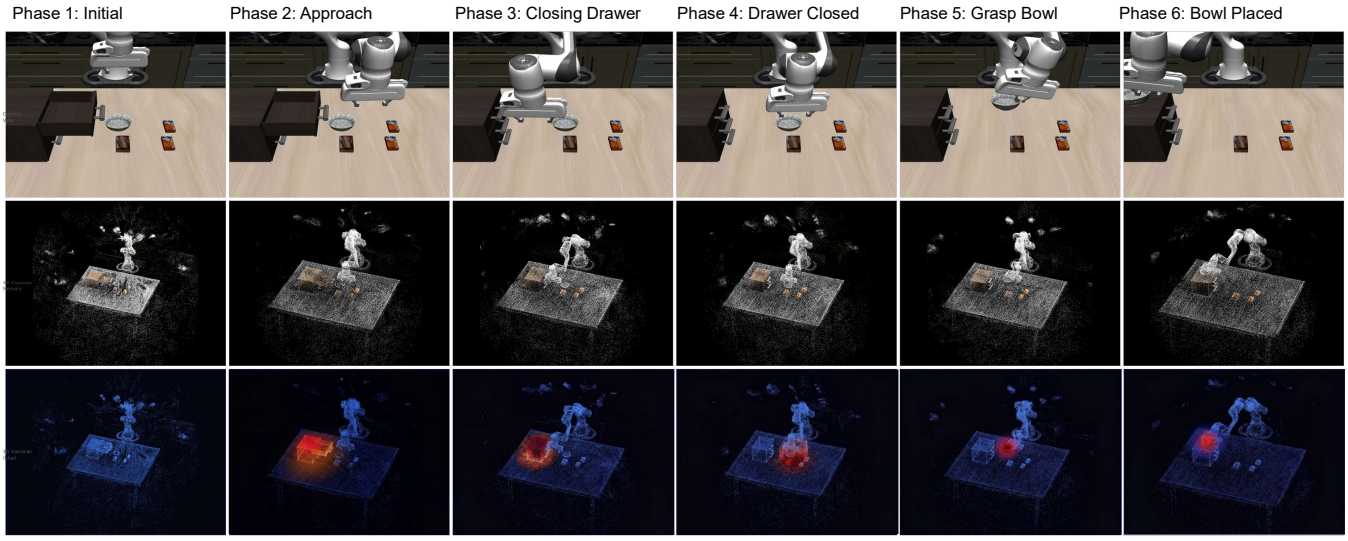

Fig. 5. **Qualitative visualization on a LIBERO Long-10 task** ("close the drawer, pick up the bowl, place it on the plate"). **Top:** RGB observations across six phases. **Middle:** Middle row shows 3D Gaussian memory rendered from the accumulated multi-step observations. **Bottom:** Learned η_k heatmap (blue = preserve, red = update aggressively). The drawer Gaussians turn red during closing, then the bowl turns red during grasping, while background remains blue—all learned from task-loss gradients without explicit programming.

(i) Evidence aggregation: $w_k = \sum_{j=1}^{N_o} A_{jk}^{\text{match}}$ measures match strength for each memory entry.

(ii) Position estimation: Observations are soft-aggregated:

$$\tilde{\boldsymbol{\mu}}_k^{\text{obs}} = \frac{\sum_j A_{jk}^{\text{match}} \, \boldsymbol{\mu}_j^{\text{obs}}}{\sum_j A_{jk}^{\text{match}} + \epsilon} \tag{9}$$

where $\boldsymbol{\mu}_j^{\text{obs}}$ is the 3D position of observation j and $\epsilon = 10^{-6}$ prevents division by zero.

(iii) Learned update gate: A learned gate converts evidence strength into an update rate:

$$\eta_k = \sigma\Big(\text{MLP}_\eta\big([\tilde{\boldsymbol{h}}_k^{\text{obs}};\, w_k;\, \text{PE}_T^{\text{rel}}(t_k^m, t)]\big)\Big) \tag{10}$$

where $\tilde{\boldsymbol{h}}_k^{\text{obs}} = \sum_j A_{jk}^{\text{match}} \, \boldsymbol{q}_j^{\text{obs},(L)} / (w_k + \epsilon)$ is the attention-weighted observation feature for memory entry k, and $\sigma(\cdot)$ constrains $\eta_k \in (0, 1)$: large values mean "trust new observation" (e.g., just-grasped object); small values mean "preserve accumulated estimate" (e.g., static background). Position updates as:

$$\boldsymbol{\mu}_k^{\text{new}} = (1 - \eta_k)\, \boldsymbol{\mu}_k^{\text{mem}} + \eta_k \, \tilde{\boldsymbol{\mu}}_k^{\text{obs}} \tag{11}$$

(iv) Existence management: Each Gaussian maintains existence probability $e_k \in (0, 1)$, updated as $e_k^{\text{new}} = \sigma(\text{logit}(e_k) + \gamma \cdot \log(w_k + \epsilon))$, where γ is a learnable coefficient. Gaussians with $e_k < \theta_e = 0.1$ are pruned.

(v) Novel object insertion: Observations with low match to all memory entries are inserted with probability $p_j^{\text{new}} = \sigma(\text{MLP}_{\text{ins}}(\boldsymbol{q}_j^{\text{obs},(L)}) - \lambda \cdot \max_k A_{jk}^{\text{match}})$, where λ is a learnable sensitivity parameter. The complete UMA data flow is illustrated in Fig. 4.

e) Why Unifying Read and Write Matters.: A two-module pipeline (separate matching for update + separate Q-Former for readout) prevents bidirectional flow. UMA's shared self-attention enables read-to-write coupling (updates bias toward task-relevant objects) and write-to-read coupling (LLM weights fresh vs. stale information). The unified gradient path $\mathcal{L}_{\text{action}} \to \mathbf{Z}^{\text{mem}} \to \text{UMA} \to \mathbf{A}^{\text{match}}, \eta_k, e_k$ enables the task loss to directly sculpt memory behavior.

E. LLM Backbone and Action Generation

We deliberately adopt the same VLM backbone as OpenVLA [9], CogACT [11], and MemoryVLA [18]—the Prismatic VLM [7], which pairs DINOv2 [14] and SigLIP [23] vision encoders with a Llama-2-7B [21] LLM—to ensure that any performance difference is attributable solely to the 3D Gaussian memory and UMA, not to backbone strength.

The LLM receives the concatenated token sequence $\mathbf{Z} = [\mathbf{Z}^{\text{mem}};\ \mathbf{Z}^{2D};\ \mathbf{Z}^{L};\ \mathbf{Z}^{S}]$, where $\mathbf{Z}^{2D}$ are visual tokens from the fused Prismatic vision encoder (DINOv2+SigLIP), $\mathbf{Z}^{L}$ are language tokens, and $\mathbf{Z}^{S}$ are proprioceptive state tokens from a 2-layer MLP. The LLM generates: (1) *Action tokens*: decoded by a diffusion-based action head into $\boldsymbol{a}_t = [\Delta \boldsymbol{p}, \Delta \boldsymbol{r}, g]$; (2) *Completion token*: a binary signal triggering the UMA update.

F. End-to-End Training

The total loss is $\mathcal{L} = \mathcal{L}_{\text{action}} + \lambda_c \mathcal{L}_{\text{complete}} + \lambda_t \mathcal{L}_{\text{temporal}} + \lambda_u \mathcal{L}_{\text{update}}$, where $\mathcal{L}_{\text{action}}$ is the standard diffusion action-prediction objective, $\mathcal{L}_{\text{complete}}$ is binary cross-entropy for sub-task completion, $\mathcal{L}_{\text{temporal}}$ encourages coherent memory for static objects via stop-gradient consistency, and $\mathcal{L}_{\text{update}}$ supervises memory positions against ground-truth with task-relevance weighting. Gradients flow through η_k and $\mathbf{A}^{\text{match}}$, directly teaching the memory how to update. Training is two-stage: pre-train FF-3DGS (frozen), then jointly train UMA, projector, and LLM (LoRA [6]) on full-length episodes (2–10 steps, curriculum).

IV. Experiments

A. Experimental Setup

a) Benchmarks: LIBERO [12]: We evaluate on all four suites—Spatial, Object, Goal and Long-10, covering 40 tasks with diverse spatial, semantic, and long-horizon challenges. Each suite contains 10 tasks evaluated over 20 trials. Results are averaged over three random seeds and then averaged across tasks.

VLABench [25]: A recent large-scale benchmark (ICCV 2025) with 100 task categories requiring world knowledge, spatial understanding, semantic reasoning, and long-horizon planning. We evaluate on Track 1 (in-distribution skill acquisition, 20 primitive tasks) and Track 6 (long-horizon composite tasks, 10 tasks), reporting success rate (SR) and progress score (PS).

b) Multi-view adaptation: LIBERO's default setup provides a single third-person camera. Since our FF-3DGS encoder requires multi-view input, we render two additional virtual viewpoints from the simulator (lateral and overhead, $\pm 30°$) at each step. The same three-view setup is used for all our LIBERO experiments. For VLABench, we use the built-in multi-camera support (front + two side cameras).

c) Baselines: For LIBERO, we compare against 10 published VLA methods with official results: **OpenVLA** [9], π_0**-FAST** [15], π_0 [2], **CogACT** [11], **MemoryVLA** [18], **SpatialVLA** [16], **TraceVLA** [27], **CoT-VLA** [26], **CronusVLA** [10], and **4D-VLA** [24]. For VLABench, we compare against π_0, π_0-FAST, and $\pi_{0.5}$ using their officially reported results. Our ablation: **Ours w/o memory**: Same Prismatic VLM backbone (Llama-2-7B + DINOv2 + SigLIP + diffusion head), no Gaussian memory—isolating the memory contribution.

d) Implementation: We use MVSplat [3] as our feed-forward 3D Gaussian Splatting (FF-3DGS) encoder to generate $M = 8192$ live Gaussians, followed by per-object pooling into $N_o \approx 20$ tokens. We use $K = 64$ readout queries, $L = 6$ UMA layers, $d = 768$, and a Prismatic VLM (Llama-2-7B, DINOv2+SigLIP). The memory cap is 32K Gaussians. MLP$_\eta$ is a 2-layer network with 256 hidden units and ReLU. Training uses AdamW with lr$= 2 \times 10^{-4}$, batch size 16, and 120K steps on 4×A100 GPUs. λ_c=1.0, λ_t=0.1, λ_u=0.5. The Prismatic backbone is identical to that used by OpenVLA, CogACT, and MemoryVLA, ensuring a controlled comparison where the only architectural difference is the Gaussian memory and UMA.

B. LIBERO Results

Table I compares GaussMemory against 10 published VLA methods. On the long-horizon suites where spatial precision matters most, GaussMemory outperforms all baselines: Goal 96.8% (vs. MemoryVLA 96.4%), Long-10 94.1% (vs. 93.4%). MemoryVLA retains an edge on Spatial/Object (98.4%) where short-horizon semantic recognition dominates. The 2D vs. 3D ablation confirms this pattern: replacing 3D Gaussians with 2D tokens yields 90.2% on Long-10—below both our 3D model and MemoryVLA—showing that explicit 3D geometry is the decisive factor.

TABLE I

LIBERO benchmark results (success rate %). †Results from MemoryVLA [18]. *Uses additional wrist-camera and proprioceptive inputs. "–" indicates not reported.

Method	Spatial	Object	Goal	Long-10
OpenVLA†	84.7	88.4	79.2	53.7
TraceVLA	84.6	85.2	75.1	54.1
SpatialVLA	88.2	89.9	78.6	55.5
CoT-VLA	87.5	91.6	87.6	69.0
CronusVLA	90.1	94.7	91.3	68.7
π_0-FAST*†	96.4	96.8	88.6	60.2
π_0^*	96.8	**98.8**	95.8	85.2
4D-VLA	93.8	92.8	95.6	86.5
CogACT†	97.2	98.0	90.2	88.8
MemoryVLA†	**98.4**	98.4	96.4	93.4
Ours w/o memory	92.4	93.6	88.2	82.4
Ours + 2D token memory	95.2	95.8	93.6	90.2
GaussMemory (3D mem.)	96.4	96.2	**96.8**	**94.1**

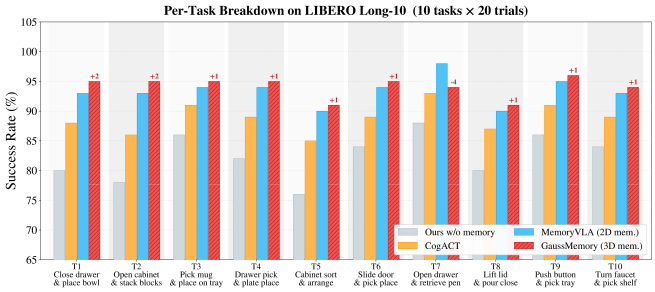

Fig. 6. **Per-task breakdown on LIBERO Long-10.** GaussMemory ranks first on 9/10 tasks. MemoryVLA leads only on T7 (semantic-heavy), consistent with the 3D-vs-2D complementarity pattern.

TABLE II

VLABench [25] results. SR: success rate (%). PS: progress score (%). ‡Official VLABench repository results.

Method	Track 1 SR	Track 1 PS	Track 6 SR	Track 6 PS
$\pi_{0.5}$-ft‡ [1]	40.6	52.4	12.0	28.6
π_0-ft‡	47.0	58.2	15.0	32.4
π_0-FAST-ft‡	51.2	62.8	18.0	36.2
Ours w/o memory	46.8	57.6	14.0	30.8
Ours + 2D token memory	51.4	63.0	19.0	38.4
GaussMemory (3D Gauss. mem.)	**56.4**	**68.2**	**24.0**	**45.6**

Memory benefit scales with task horizon. The gap over w/o memory grows monotonically: +4.0 (Spatial) → +8.6 (Goal) → +11.7 (Long-10), confirming that persistent 3D memory becomes increasingly critical as tasks accumulate more state changes. Fig. 6 shows GaussMemory ranks first on 9/10 individual Long-10 tasks.

C. VLABench Results

Table II evaluates GaussMemory on VLABench. On Track 1, GaussMemory achieves 56.4% SR, outperforming π_0-FAST [15] (51.2%) by +5.2%. The advantage is larger on Track 6 (long-horizon composite): 24.0% vs. 18.0% (+6.0%). Notably, the **2D vs. 3D memory ablation** isolates the rep-

TABLE III

ABLATION STUDY ON LIBERO LONG-10 (SUCCESS RATE %).

Configuration	Long-10	Δ
Prismatic VLA backbone (no memory)	82.4	–
Passive → Active memory		
+ 3D Gaussian Memory (fixed η=0.3)	90.4	+8.0
+ learned η (active update gate)	92.8	+10.4
+ MLP-modulated η (full active gate)	93.2	+10.8
+ soft existence management	**94.1**	**+11.7**
Unified vs. Decoupled read/write		
Decoupled (separate match + readout)	91.8	+9.4
UMA (unified read/write, ours)	**94.1**	**+11.7**
View count		
2 views	91.4	+9.0
3 views (default)	**94.1**	**+11.7**
Memory capacity		
4K Gaussians	78.2	−4.2
8K Gaussians	86.4	+4.0
16K Gaussians	91.8	+9.4
32K Gaussians (default)	**94.1**	**+11.7**
64K Gaussians	93.6	+11.2

resentation effect: replacing our 3D Gaussian memory with a 2D token buffer (same UMA architecture, same backbone) yields 51.4% / 19.0% on Track 1/6—substantially below the full 3D model (56.4% / 24.0%) and only marginally above π_0-FAST. This confirms that the 3D geometric representation, not merely the presence of memory, is the key contributor.

VLABench amplifies the memory advantage. VLABench tasks have longer horizons (500+ timesteps) than LIBERO (∼120). The improvement over π_0-FAST grows from +5.2% (Track 1) to +6.0% (Track 6), mirroring the "memory scales with horizon" pattern from LIBERO.

Cross-benchmark consistency. Across both benchmarks, three patterns hold: (1) memory is the single largest factor; (2) the advantage grows with task horizon; (3) it is most pronounced on spatially demanding tasks. This consistency across fundamentally different task designs supports the generality of active 3D Gaussian memory.

Table III reveals how each component contributes on LIBERO Long-10:

3D Gaussian memory is the foundation (passive memory): Simply adding persistent 3D Gaussian memory with a fixed update rate (η=0.3) already yields +8.0 (82.4→90.4), the single largest gain. This "passive" memory treats every object equally—confirming that 3D geometric representation itself is the primary contributor.

Passive→Active: learned update strategy: Replacing fixed η with attention-derived w_k adds +2.4 (90.4→92.8)—this is the transition from *passive* (update everything equally) to *active* (update task-relevant objects more). The MLP gate (+0.4) and soft existence management (+0.9) further sharpen the active strategy, bringing the total to +11.7. The emergent η hierarchy (0.74 for manipulated targets, 0.06 for background) is the direct outcome of this learned active update (Fig. 9).

Unified vs. Decoupled: A decoupled baseline (separate matching + separate Q-Former readout, matched parameters) achieves 91.8%, 2.3 points below UMA (94.1%)—confirming that unified read/write coupling itself contributes beyond the 3D representation alone.

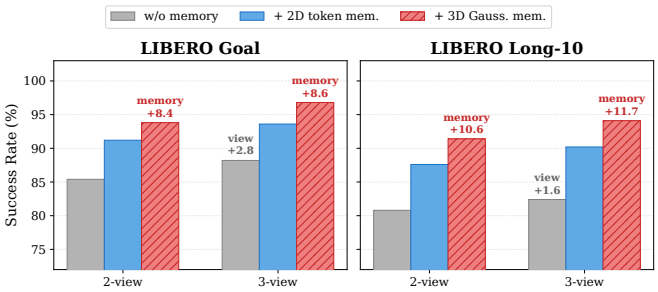

Fig. 7. **Multi-view fairness.** At every view count, the memory contribution (red annotation) vastly exceeds the view contribution (gray annotation), confirming gains come from 3D memory, not extra cameras.

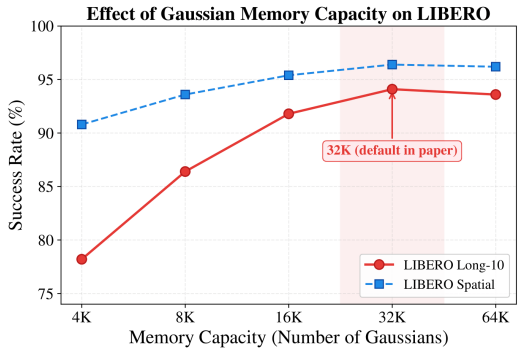

Fig. 8. **Gaussian memory capacity ablation.** Performance scales steeply up to 32K (our default) then plateaus, indicating effective learned pruning.

View count robustness (Fig. 7): The memory contribution far exceeds the view-count contribution. At 3-view, memory adds +11.7 on Long-10 while adding a third view (2→3) only adds +1.6 without memory—confirming gains come from 3D memory, not extra cameras.

Memory capacity (Fig. 8): Performance scales with Gaussian count up to 32K, beyond which marginal returns diminish (64K: 93.6%). The steep rise from 4K to 16K indicates sufficient capacity is critical; the plateau at 32K→64K confirms the learned existence management prevents redundant Gaussians.

D. Emergent Memory Behavior

Fig. 5 and Fig. 9 visualize the emergent update strategy: manipulated objects receive $\eta \approx 0.74$ while background stays at $\eta \approx 0.06$-the update strategy emerges without handcrafted update rules, driven by task-level action supervision and auxiliary memory-consistency losses. Fig. 10 shows the memory size dynamics: insertion spikes during scene changes while pruning maintains equilibrium well below the 32K cap.

V. CONCLUSION

We presented GaussMemory, which shifts robot spatial memory from passive recording to active, task-driven learn-

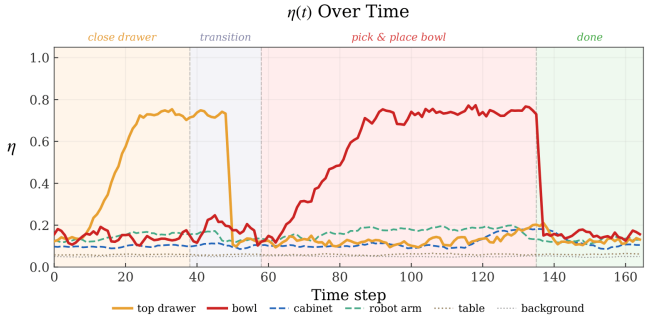

Fig. 9. **Temporal η_k on a LIBERO Long-10 episode.** The drawer's η peaks during closing, then the bowl's η rises during grasping. Background maintains $\eta \approx 0.06$ throughout—learned through end-to-end gradient flow.

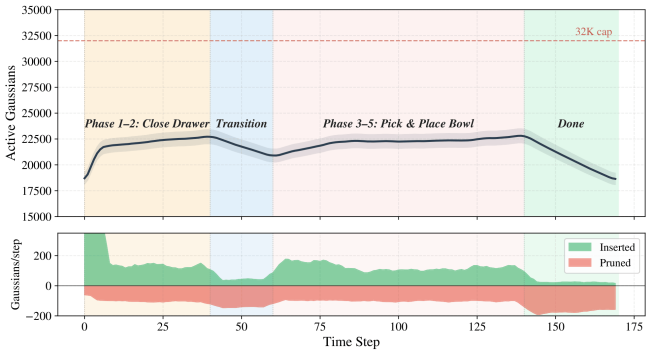

Fig. 10. **Memory size dynamics** on a LIBERO Long-10 episode. **Top:** active Gaussian count stays well below the 32K cap. **Bottom:** insertion (green) spikes during scene changes; pruning (red) maintains equilibrium.

ing via 3D Gaussian Splatting and UMA. The system learns *what matters*—manipulated objects are updated aggressively while background is left untouched—all discovered automatically from task performance. On LIBERO, GaussMemory outperforms MemoryVLA on Goal and Long-10.

References

[1] Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y Galliker, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. In *9th Annual Conference on Robot Learning*, 2025.

[2] 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.

[3] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. MVSplat: Efficient 3D Gaussian Splatting from sparse multi-view images. In *ECCV*, 2024.

[4] Qiao Gu, Alihusein Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Anoop Cherian, et al. ConceptGraphs: Open-vocabulary 3D scene graphs for perception and planning. In *ICRA*, 2024.

[5] Anna-Maria Halacheva, Jan-Nico Zaech, Xi Wang, Danda Pani Paudel, and Luc Van Gool. Gaussianvlm: Scene-centric 3d vision-language models using language-aligned gaussian splats for embodied reasoning and beyond. *IEEE Robotics and Automation Letters*, 2025.

[6] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

[7] 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.

[8] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for real-time radiance field rendering. In *SIGGRAPH*, 2023.

[9] 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. In *CoRL*, 2024.

[10] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 18388–18396, 2026.

[11] 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.

[12] 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*, 2023.

[13] Guanxing Lu, Shiyi Zhang, Ziwei Wang, Changliu Liu, Jiwen Lu, and Yansong Tang. ManiGaussian: Dynamic Gaussian Splatting for multi-task robotic manipulation. In *ECCV*, 2024.

[14] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *TMLR*, 2024.

[15] 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.

[16] 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.

[17] Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian Reid, and Niko Suenderhauf. SayPlan: Grounding large language models using 3D scene graphs for scalable robot task planning. In *CoRL*, 2023.

[18] 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.

[19] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

[20] Marcel Torne et al. MEM: Multi-scale embodied memory for vision-language-action models. *arXiv preprint arXiv:2603.03596*, 2026.

[21] 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.

[22] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3D scenes. In *ECCV*, 2024.

[23] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

[24] 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*, 2025.

[25] Shiduo Zhang, Zhe Xu, Peiju Liu, et al. VLABench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *ICCV*, 2025.

[26] 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*, 2025.

[27] 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 *International Conference on Learning Representations*, 2025.