Global Prior Meets Local Consistency: Dual-Memory Augmented Vision-Language-Action Model for Efficient Robotic Manipulation

Zaijing Li[1 2], Bing Hu[1], Rui Shao[1 3*], Gongwei Chen[1], Dongmei Jiang[2*],
Pengwei Xie[4], Jianye HAO[4], Liqiang Nie[1]
[1]Harbin Institute of Technology, Shenzhen [2]PengCheng Laboratory
[3]Shenzhen Loop Area Institute [4]Huawei Noah's Ark Lab
https://cybertronagent.github.io/OptimusVLA.github.io/

Abstract

*Hierarchical Vision–Language–Action (VLA) models have rapidly become a dominant paradigm for robotic manipulation. It typically comprising a Vision–Language backbone for perception and understanding, together with a generative policy for action generation. However, its performance is increasingly bottlenecked by the action generation proceess. (i) Low inference efficiency. A pronounced distributional gap between isotropic noise priors and target action distributions, which increases denoising steps and the incidence of infeasible samples. (ii) Poor robustness. Existing policies condition solely on the current observation, neglecting the constraint of history sequence and thus lacking awareness of task progress and temporal consistency. To address these issues, we introduce **OptimusVLA**, a dual-memory VLA framework with **G**lobal **P**rior **M**emory (**GPM**) and **L**ocal **C**onsistency **M**emory (**LCM**). GPM replaces Gaussian noise with task-level priors retrieved from semantically similar trajectories, thereby shortening the generative path and reducing the number of function evaluations (NFE). LCM dynamically models executed action sequence to infer task progress and injects a learned consistency constraint that enforces temporal coherence and smoothness of trajectory. Across three simulation benchmarks, OptimusVLA consistently outperforms strong baselines: it achieves 98.6% average success rate on LIBERO, improves over π_0 by 13.5% on CALVIN, and attains 38% average success rate on RoboTwin 2.0 Hard. In Real-World evaluation, OptimusVLA ranks best on Generalization and Long-horizon suites, surpassing π_0 by 42.9% and 52.4%, respectively, while delivering 2.9× inference speedup.*

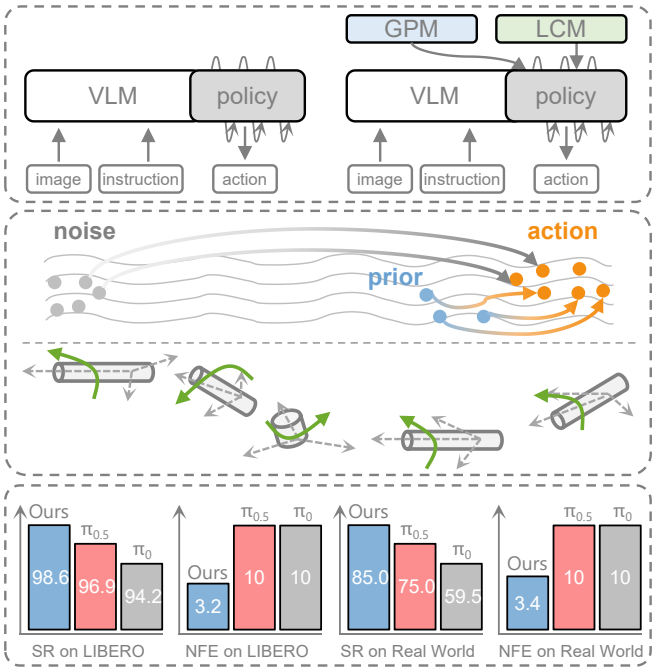


Figure 1. **Top**: Comparison between the standard VLA architecture (left) and our proposed OptimusVLA (right). (ii) Poor robustness to temporal dependence. **Middle**: Illustration of how GPM (blue) and LCM (green) address two key limitations of existing VLA models: (i) Low inference efficiency due to a large prior–target gap. (ii) Poor robustness to temporal dependence. **Bottom**: Efficiency and performance comparison.

1. Introduction

Vision–Language–Action (VLA) models [1, 12, 14, 18, 19, 37, 40, 46, 56, 60] are rapidly emerging as a dominant paradigm for robotic manipulation [13, 20, 39]. As shown in Fig. 1 Top, it integrates observation perception, natural-language understanding, and action generation within an end-to-end framework. Leveraging advances in vision–language models [4, 27] and generative policies [6, 16, 54], VLA models have demonstrated strong performance across a variety of benchmarks [5, 26, 34]. However,

the action generation process has become a dominant bottleneck in terms of both efficiency and robustness.

Limitation I: Inefficient action generation due to a large prior–target gap. As shown in Fig. 1 Middle, mainstream VLA models map a prior noise distribution (typically Gaussian) to the target action distribution via diffusion denoising or flow matching [12, 36]. However, the cross-domain transformation from isotropic noise to structured actions is large, necessitating multi-steps denoising to reach high(-quality actions [32]. Moreover, the stochastic starting point frequently initializes the generative process in regions that kinematically invalid actions [33, 35]. While a naive strategy might use *action-prior* as starting point, this approach severely curtails diversity, collapsing the learned mapping to a restrictive *similar-to-target* function and failing to address how such priors are obtained.

Limitation II: Poor robustness to temporal dependence. VLA models [12, 14] rely solely on the current observation often fails to distinguish between distinct task phases that yield similar visual inputs (e.g., distinguishing an unopened drawer from one that has just been closed). Moreover, lacking temporal dependency weakens consistency with the executed trajectory, leading to jittery control. While a straightforward solution [8, 29] is to concatenate long-sequence historical observation to the input, it substantially increases inference overhead and memory usage. Moreover, it misaligned the pre-training distribution of VLA models [38].

To address these issues, we propose **OptimusVLA**, a VLA framework jointly driven by **G**lobal **P**rior **M**emory (**GPM**) and **L**ocal **C**onsistency **M**emory (**LCM**). **GPM** replaces isotropic noise with task-level priors as the generative start, narrowing the *prior–target* distributional gap and reducing required number of function evaluations (NFE). **LCM** dynamically models recent action sequences to infer task progress and injects a consistency constraint that smooths controls. Together, GPM and LCM improve efficiency and robustness of OptimusVLA through global prior alignment and local consistency constraints.

The core innovation of GPM is to treat prior initialization as a memory-driven retrieval problem rather than a fixed noise design. It consists of three components: a Prior Head that integrates current multimodal information; a Memory Bank that stores and retrieves task-level priors; and a Prior-Aware Sampler for prior action distribution sampling together with adaptive noise scale and NFE. By initializing the generative flow in the neighborhood of similar tasks, GPM drastically narrows the prior-target gap. This not only reduce the required NFE but, critically, anchors the process in feasible action space. Adaptive noise injection retains the model's exploratory power, preventing deterministic collapse and preserving generalization to novel tasks.

Concurrently, the LCM endows the VLA model with temporal perception without the burden of long context modeling. It employs two lightweight structures: a Consistency Layer that ingests recent action chunks to model local consistency, and a Dynamic-Awareness module that dynamicaly models historical sequence to infer task progress. The LCM learns a consistency constraint that injected into the policy input, without requiring any modification to the VLA pre-training paradigm. It enforces consistency with historical trajectory, and provides crucial progress awareness with negligible computational overhead.

We conduct comprehensive evaluation across three simulators and the real world. On LIBERO [26], OptimusVLA attains an average success rate of 98.6%, surpassing current SOTA baselines (e.g., $\pi_{0.5}$ [12] and MemoryVLA [38]). On CALVIN [34], it improves over π_0 by 13.5%, and it achieves 38% average success rate on RoboTwin 2.0 [5] *Hard* setting. In real-world evaluations, OptimusVLA ranks best on both the *Generalization* and *Long-horizon* suites, outperforming π_0 by 42.9% and 52.4%, respectively, while delivering 2.9× inference speedup. These results indicate that memory-driven prior initialization and temporal constraint can boost robustness of VLA models without sacrificing efficiency. Our contributions are as follows:

- A novel **G**lobal **P**rior **M**emory that narrows the prior-target action distribution gap. By replacing isotropic noise with retrieved task-level prior, GPM significantly reduces NFE and lowers the risk of infeasible sampling.
- A lightweight **L**ocal **C**onsistency **M**emory that endows VLAs with temporal awareness. LCM dynamically models recent actions to inject a consistency constraint, providing progress-awareness without appreciable computational overhead.
- We propose a dual-memory VLA framework, OptimusVLA, which driven by GPM and LCM. Extensive experiments results across 3 simulators and real-world show that OptimusVLA achieves higher performance with substantially inference speedup.

2. Preliminaries and Related Work

Contemporary Vision–Language–Action (VLA) models typically divided into two categories: single-stream architectures [14, 15, 30, 37, 61] and hierarchical architectures [1, 7, 12, 17, 36]. Single-stream architectures [23, 48, 59] employ a vision–language backbone and autoregressively generate discretized action tokens [11, 13]. To meet the high-frequency control demands of real-world scenarios, various efficiency techniques, such as parallel decoding [15, 40], parameter quantization [45], and token compression [18, 20], have been adopted to increase inference speed. On the other hand, hierarchical approaches [22, 24, 55] contain a vision–language backbone for high-level, low-frequency planning, a generative policy [6, 54] for low-level, high-frequency action generation [1, 12, 47]. This decomposition substantially accelerates inference compared

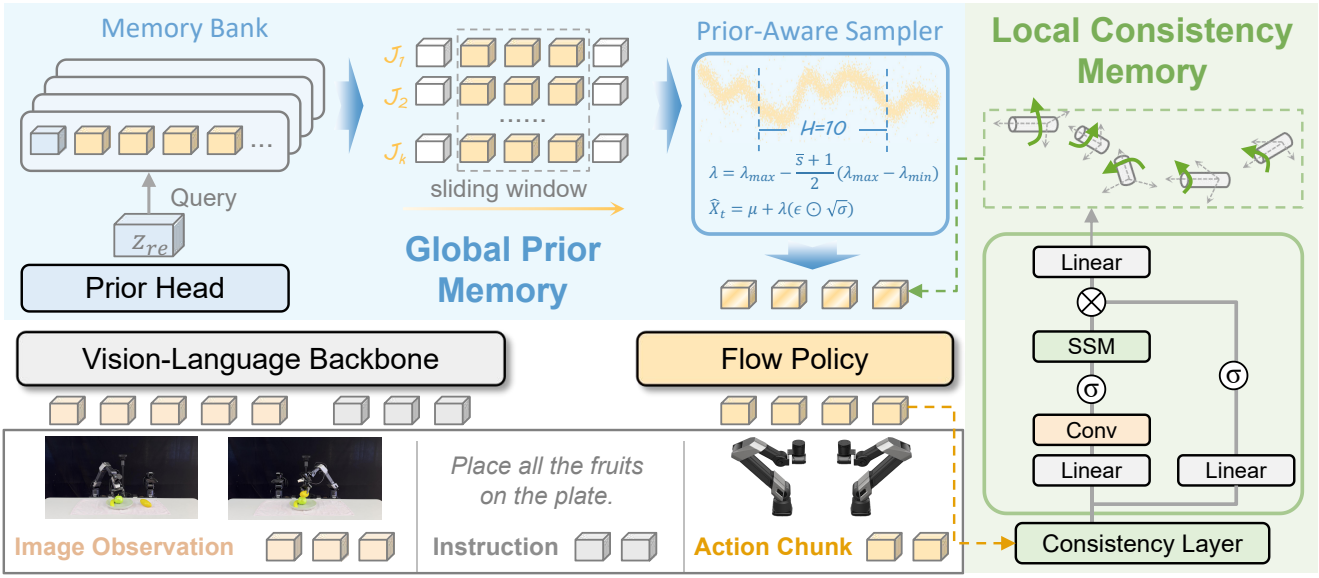


Figure 2. Overview of OptimusVLA framework. Given a task and the current observation, the Vision–Language backbone first encodes the inputs into a multimodal representation. GPM then retrieves a task-level prior based on this representation, while LBM dynamically encodes the historical action sequence to produce a consistency constraint. Finally, the flow policy denoises the initialization with an adaptive NFEs schedule to generate the action chunk.

with purely autoregressive pipelines. Nevertheless, even with flow matching [25, 32], the policy typically requires multiple NFEs to reach high quality actions, which ultimately caps the achievable speedup.

Conditional Flow Matching (CFM) trains a time-conditioned velocity field $v_\theta(x, t)$ that transports the noise $x_0 \sim \mathcal{P}_0$ to the target action $x_1 \sim \mathcal{P}_1$. A common and stable approach is to define an Optimal Transport (OT) path as a straight line:

$$x_t = (1-t)x_0 + tx_1. \tag{1}$$

The target velocity u_t along this path is constant, given by $u_t(x_t \mid x_0, x_1) = x_1 - x_0$. The learning objective is to train v_θ to match this target velocity field:

$$\min_\theta \ \mathbb{E}_{t\sim\mathcal{U}[0,1], x\sim p_t(x)} \left\| v_\theta(t,x) - u_t(x) \right\|_2^2, \tag{2}$$

When $\mathcal{P}_0$ is isotropic Gaussian distribution and $\mathcal{P}_1$ is a structured action distribution, the source-target gap can be large, which in practice yields more number of function evaluations (NFE) to reach high-quality actions. To narrow the gap, we introduce Global Prior Memory, which replaces the Gaussian source by a task-level prior constructed from semantically similar action trajectories.

On the other hand, mainstream work [1, 12, 14] formulates action generation in VLA as a Markov Decision Process, conditioning the model solely on the current observation while ignoring the historical sequence. This Markovian assumption hampers phase awareness: the agent cannot discern task progress from visually similar states (e.g., whether the drawer has not yet been opened or has already been closed), leading to suboptimal or inconsistent behaviors [38]. To provide temporal context, some work [8, 29] adapt the VLM paradigm to sequence modeling, explicitly encoding long observation streams. While effective, these approaches substantially increase latency and memory usage. Other methods simply concatenate past states [58] or actions [3] to the input, which lacks a global representation of long-range temporal context [21, 50–52]. Moreover, it misaligned the VLA model's single-frame pre-training distribution [14]. A different line of work employs working memory [24, 38] to dynamically model observation history. However, these designs typically rely on the VLM backbone to represent the current multimodal information at every step; consequently, each memory update incurs a full VLM forward pass, creating a throughput bottleneck for fast control loops. In this paper, we propose Local Consistency Memory (LCM), a lightweight working memory that dynamically encodes the recent sequence with minimal inference overhead. LCM equips VLA policies with explicit progress awareness and temporal consistency without repeatedly invoking heavy VLM computations, thereby preserving real-time efficiency.

3. OptimusVLA

In this section, we first give an overview of our proposed OptimusVLA. Next, we introduce the details of **G**lobal **P**rior **M**emory (**GPM**) in Sec. 3.2. Subsequently, we elaborate on how to implement the proposed **L**ocal **C**onsistency

Memory (**LCM**) (Sec. 3.3). Finally, the training strategy of the OptimusVLA is explained in Sec 3.4.

3.1. Overview of OptimusVLA

As shown in Figure 2, OptimusVLA comprises Vision–Language backbone [4, 27], flow policy [54], Global Prior Memory (GPM), and Local Consistency Memory (LCM). The Vision–Language backbone and flow policy are the standard components of a hierarchical VLA architecture. GPM explicitly tackles the distributional gap between the standard noise prior and the structured action distribution, while LCM injects progress-aware temporal constraints, jointly enhancing efficiency and robustness of action generation.

Formally, given a natural-language instruction ℓ and the current observation O_t, the multimodal representation E_{emb} is embedded by Vision-Language backbone `VLM`:

$$E_{emb} \leftarrow \texttt{VLM}(O_t, \ell) \tag{3}$$

where O_t typically consists of the proprioceptive state q_t, and multi-view images $I_t^1, I_t^2, \ldots, I_t^n$.

Subsequently, the multimodal representation E_{emb} is projected into a retrieval token z_{re}, which queries the memory bank in GPM to obtain a task-level prior action distribution $\mathcal{P}_{re}$:

$$\mathcal{P}_{re} \leftarrow \texttt{GPM}(z_{re}) \tag{4}$$

At time t, the GPM samples an action chunk $\hat{\mathbf{X}}_t \in \mathbb{R}^{H\times A} \sim \mathcal{P}_{re}$ with adaptive noise and NFE, where H is the chunk length and A is the action dimension. Concurrently, the LCM processes the last action chunk $\mathbf{A}_{t-1}$ to yield a consistency bias $\mathbf{B}_t \in \mathbb{R}^{H\times A}$, then this bias is injected to form the final policy input $\mathbf{X}_t$:

$$\mathbf{B}_t \leftarrow \texttt{LCM}(\mathbf{A}_{t-1}) \tag{5}$$

$$\mathbf{X}_t = \hat{\mathbf{X}}_t + \mathbf{B}_t \tag{6}$$

Finally, the flow policy p_θ transforms $\mathbf{X}_t$ into action chunk $a_{t+1:t+H}$ based on adaptive NFE N:

$$a_{t+1:t+H} \leftarrow p_\theta(\mathbf{X}_t, N) \tag{7}$$

3.2. Global Prior Memory

Standard flow matching transports isotropic Gaussian noise to the action space [25, 32]. Due to the large discrepancy between these distributions, this process requires many NFEs and risks collapsing to infeasible actions. In robotics, however, similar tasks share related action distributions (e.g., `pick_a_cup` and `pick_a_plate`). Motivated by this, our **G**lobal **P**rior **M**emory (**GPM**) stores, retrieves, and composes trajectories into task-level priors. This fundamentally relocates the generative starting point from $\mathcal{N}(0, I)$ to the neighborhood of the target manifold, thereby narrowing the prior-target gap.

GPM is a long-term memory module composed of a Prior Head, a Memory Bank, and a Prior-Aware Sampler.

Prior Head. A lightweight MLP projects the multimodal representation E_{emb} into a retrieval token z_{re}:

$$z_{re} = PriorHead(E_{emb}) \tag{8}$$

Memory Bank. z_{re} queries the Memory Bank, which stores M key-value pairs $\{z_m, J_m\}_{m=1}^M$ of task embeddings and their corresponding full trajectories. We compute cosine similarities and retrieve the k nearest trajectories $\{J_i\}_{i=1}^k$ and their scores s_i:

$$\{J_i, s_i\}_{i=1}^k \leftarrow \text{MemoryBank}(z_{re}) \tag{9}$$

Then we compute a normalized global similarity $\bar{s}$ and softmax weights α_i:

$$\alpha_i = \text{softmax}(s_i/\tau_s), \quad \bar{s} = \sum_{i=1}^k \alpha_i s_i, \tag{10}$$

where τ_s is a temperature. For each retrieved trajectory J_i, we extract action blocks $C_i \in \mathbb{R}^{H\times A}$ based on a sliding window. The task-level prior $\mathcal{P}_{re} = \mathcal{N}(\mu, \text{diag}(\text{Var}))$ is then formed by a weighted average:

$$\mu = \sum_{i=1}^k \alpha_i C_i, \qquad \text{Var} = \sum_{i=1}^k \alpha_i \left(C_i - \mu\right)^{\odot 2}, \tag{11}$$

where $\odot 2$ denotes element-wise power.

Prior-Aware Sampler. Subsequently, Prior-Aware Sampler samples the initialization with similarity-adaptive noise λ, and adaptive NFE N:

$$\lambda = \lambda_{\max} - \tfrac{\bar{s}+1}{2}\left(\lambda_{\max} - \lambda_{\min}\right), \tag{12}$$

$$N = N_{\min} + \left(1 - \tfrac{\bar{s}+1}{2}\right)\left(N_{\max} - N_{\min}\right), \tag{13}$$

$$\hat{\mathbf{X}}_t = \mu + \lambda\left(\epsilon \odot \sqrt{\text{Var}}\right), \qquad \epsilon \sim \mathcal{N}(0, I), \tag{14}$$

where $\lambda_{\min}$, $\lambda_{\max}$, $N_{\min}$, $N_{\max}$ are hyper-parameters.

GPM draws the prior distribution into the vicinity of the target distribution, thereby drastically curtailing the required NFEs and improving the inference efficiency of the flow policy. Moreover, by initializing from a reliable prior, it reduces the likelihood that sampled actions fall into infeasible regions.

3.3. Local Consistency Memory

Standard VLAs, adhering to the Markovian assumption, lack explicit progress awareness [38] and suffer from jittery control. Our **L**ocal **C**onsistency **M**emory (LCM) addresses this by dynamically modeling the action history to infer task progress and enforce temporal consistency, all with minimal computational overhead. LCM serves as a working

Table 1. Performance comparison on LIBERO [26]. We report the average success rate on each task suite (500 rollouts).

Method	LIBERO				
	Spatial	Object	Goal	Long	Avg.
DP [6]	78.3	92.5	68.3	50.5	72.4
Octo [42]	78.9	85.7	84.6	51.1	75.1
SpatialVLA [37]	88.2	89.9	78.6	55.5	78.1
π_0-FAST [36]	96.4	96.8	88.6	60.2	85.5
CogACT [17]	97.2	98.0	90.2	88.8	93.6
MemoryVLA [38]	98.4	98.4	96.4	93.4	96.7
OpenVLA-OFT [15]	97.6	98.4	97.9	94.5	97.1
OpenVLA [14]	84.7	88.4	79.2	53.7	76.5
UniVLA [3]	95.4	98.8	93.6	94.0	95.4
π_0 [1]	96.8	98.8	95.8	85.2	94.2
$\pi_{0.5}$ [12]	98.8	98.2	98.0	92.4	96.9
OptimusVLA	**99.6**	**99.8**	**98.4**	**96.4**	**98.6**

Table 2. Performance comparison on CALVIN [34]. We report the success rate of each track and average completion length (Avg. Len). † represents the result we reproduced.

Method	CALVIN (ABC → D)					
	1/5	2/5	3/5	4/5	5/5	Avg. Len
RoboVLM [28]	**98.0**	**93.6**	85.4	77.8	70.4	4.25
ReconVLA [41]	95.6	87.6	76.9	69.3	64.1	3.95
OpenVLA [14]	91.3	77.8	62.0	52.1	43.5	3.27
UniVLA [3]	95.5	85.8	75.4	66.9	56.5	3.80
UP-VLA [53]	92.8	86.5	81.5	76.9	69.9	4.08
RoboDual [2]	94.4	82.7	72.1	62.4	54.4	3.66
Seer [43]	96.3	91.6	86.1	80.3	74.0	4.28
VPP [10]	95.7	91.2	86.3	81.0	75.0	4.29
π_0 [1]	93.8	85.0	76.7	68.1	59.9	3.92
$\pi_{0.5}$† [12]	94.4	88.4	85.3	80.1	76.1	4.26
OptimusVLA	97.6	93.2	**88.8**	**85.7**	**78.1**	**4.45**

memory, comprising a Consistency Layer and a Dynamic Awareness Module.

Consistency Layer. At time step t, the Consistency Layer incorporates the previous action chunk $\mathbf{A}_{t-1} = [\mathbf{a}_{t-H+1}, \dots, \mathbf{a}_t] \in \mathbb{R}^{H\times A}$ and employs a self-attention mechanism [44] to capture inter-action dependencies and constraints within the chunk:

$$\hat{\mathbf{B}}_{t-1} \leftarrow ConsistencyLayer(\mathbf{A}_{t-1}) \tag{15}$$

Dynamic Awareness Module. This module is designed to capture inter-chunk temporal dynamics. It takes the representation $\hat{\mathbf{B}}_{t-1}$ and updates its internal state to predict the consistency bias $\mathbf{B}_t$ for the next time step (as in Eq. 5). We implement this module using a Mamba-based structure [9], which efficiently models long-range dependencies with linear complexity:

$$\mathbf{B}_t \leftarrow DynamicAwareness(\hat{\mathbf{B}}_{t-1}) \tag{16}$$

LCM focuses on the temporal dependencies of the current trajectory, converting consistency in the action flow into an additive constraint, thereby yielding smoother sampled trajectories.

3.4. Training Details

The training pipeline consists of three stages. First, we pretrain a hierarchical VLA model based on the architecture and training protocol of $\pi_{0.5}$ [12]. Next, we train the Prior Head to learn task-discriminative representations from the batch via InfoNCE objective:

$$\mathcal{L}_{\text{GPM}} = -\mathbb{E}_q \left[\log \frac{\exp(\text{sim}(z_{re}, z^+)/\tau_c)}{\sum_{j\in\mathcal{N}(q)} \exp(\text{sim}(z_{re}, z_j)/\tau_c)} \right] \tag{17}$$

where $\text{sim}(\cdot)$ denotes cosine similarity, τ_c is a temperature parameter, and $\mathcal{N}(q)$ denotes the set of in-batch negative samples. Finally, we freeze GPM and train LCM to predict the residual required to bridge the gap between the global prior mean μ_t and the ground-truth action chunk $\mathbf{A}_t^\star$:

$$\mathbf{B}_t^\star = \mathbf{A}_t^\star - \mu_t \tag{18}$$

$$\mathcal{L}_{\text{LCM}} = \mathbb{E}_{(\mathbf{A}_{t-1}, \mathbf{A}_t^\star, \mu_t)\sim\mathcal{D}} \left[\|\mathbf{B}_t - \mathbf{B}_t^\star\|_2^2 \right] \tag{19}$$

where $\mathbf{A}_t^\star$ and μ_t are the ground-truth action chunk and mean prior at timme t. More details are in **Appendix**.

4. Experiments

We conduct comprehensive experiments to validate OptimusVLA, focusing on three pivotal research questions: **Q1**. Does OptimusVLA outperform state-of-the-art VLA models on diverse simulation benchmarks? **Q2**. Does OptimusVLA demonstrate superior efficiency and performance in real-world? **Q3**. How do GPM and LCM individually contribute to the performance of OptimusVLA?

4.1. Evaluation on Simulation Benchmarks

Simulation Benchmarks. To evaluate the robustness of OptimusVLA across diverse simulation environments, we conduct experiments on LIBERO [26], CALVIN [34], and RoboTwin 2.0 [5]. LIBERO is organized into four task suites (Spatial, Object, Goal, and Long) with each contains 10 tasks. We report the average success rate in each suite with 500 rollouts. CALVIN consists of 4 distinct environments, and we train on 3 environments and testing on the held-out one (ABC → D). We report the success rate for each track, as well as the average length over 5 tasks, using 500 rollouts. On RoboTwin 2.0, we randomly select 16

Table 3. Performance comparison on RoboTwin 2.0 [5]. We report per-task success rates (SR) and rank (RK) over 100 rollouts under *Hard* setting. † represents the result we reproduced. 16 tasks results are available in Appendix.

Task	RDT [31]		ACT [57]		DP [6]		DP3 [49]		π_0 [1]		$\pi_{0.5}$†[12]		OptimusVLA	
	SR	RK	SR	RK	SR	RK	SR	RK	SR	RK	SR	RK	SR	RK
Click Alarmclock	12%	4	4%	7	5%	6	14%	3	11%	5	18%	2	31%	1
Click Bell	9%	3	3%	4	0%	5	0%	5	3%	4	28%	2	46%	1
Dump Bin Bigbin	32%	3	1%	6	0%	7	53%	1	24%	5	29%	4	35%	2
Open Laptop	32%	4	0%	6	0%	6	7%	5	46%	2	38%	3	48%	1
Place Bread Skillet	1%	3	0%	4	0%	4	0%	4	1%	4	2%	2	4%	1
Place Container Plate	17%	4	1%	5	0%	6	1%	5	45%	1	30%	3	37%	2
Press Stapler	24%	4	6%	5	0%	7	3%	6	29%	3	36%	2	45%	1
Stack Bowls Two	30%	4	0%	6	0%	6	6%	5	41%	3	49%	2	58%	1
Average	20%	4	2%	6	1%	7	11%	5	25%	3	29%	2	38%	1

Table 4. Ablation study of GPM and LCM on LIBERO-Long, CALVIN, and Real-World *Generalization Tasks*.

Ablation Setting		Simulation		Real-World
GPM	LCM	LIBERO-Long	CALVIN	*Generalization Tasks*
✓	✓	**96.4**	**4.45**	**85.0**
	✓	93.2 (↓ 3.3%)	4.28 (↓ 3.8%)	77.0 (↓ 9.4%)
✓		94.8 (↓ 1.7%)	4.38 (↓ 1.6%)	79.5 (↓ 6.5%)
		92.4 (↓ 4.1%)	4.26 (↓ 4.3%)	75.0 (↓ 11.8%)

Table 5. Ablation study on memory size of GPM. We report average success rate on LIBERO-Long. `Num.` denotes the number of trajectories in the memory bank, `k.` denotes the number of retrieved trajectories.

Metric	Num=6500			Num=1300		Num=130
	k=8	k=16	k=1	k=8	k=1	k=8
SR	**96.4**	94.8	92.6	95.2	92.4	93.6

tasks for evaluation, and test the models with 100 rollouts under the *Hard* (domain-randomized with clutter, lighting, textures, and height variations) settings.

Implementation Details. OptimusVLA is initialized from the weights of $\pi_{0.5}$ [12], and then augment it with the proposed GPM and LCM modules, resulting in 3.6B parameters in total. We train OptimusVLA on 8× NVIDIA A800 GPUs with a global batch size of 512 for 30,000 steps. The learning rates on LIBERO, CALVIN, and RoboTwin are set to 5e-5. More details are provided in the **Appendix**. in

Results on LIBERO. As shown in Table 1, OptimusVLA achieves an average success rate of 98.6% on LIBERO, surpassing existing state-of-the-art methods, including $\pi_{0.5}$ [12] and OpenVLA-OFT [15]. In particular, most of VLA models struggle on the LIBERO-Long suite due to error accumulation over long sequences. OptimusVLA mitigates this via GPM, which retrieves task-specific priors to anchor the generative process. By initializing the flow closer to the target manifold, GPM not only stabilizes long-horizon generation but also drastically reduces the NFEs (3.2 for OptimusVLA vs. 10.0 for $\pi_{0.5}$)

Results on CALVIN. As shown in Table 2, OptimusVLA attains an average episode length of 4.45 on CALVIN, outperforming π_0 [1] by 13.5% and surpassing $\pi_{0.5}$ [12]. Under the ABC → D setting, π_0 relies on task-agnostic Gaussian noise, which is brittle to the distribution shifts inherent in unseen environments. In contrast, OptimusVLA leverages GPM to retrieve semantically similar trajectories from training data as initialization. This *semantic anchoring* allows the model to adapt to novel scenes by deforming a viable action prior rather than generating from scratch, thereby exhibiting superior zero-shot robustness.

Results on RoboTwin 2.0. As shown in Table 3, OptimusVLA achieves the average success rate of 38% on RoboTwin 2.0 *Hard* setting. In particular, it attains a 58% success rate on the *Stack Bowls Two* task, outperforming RDT [31] by +28% success rate. Bimanual manipulation demands high temporal and inter-arm consistency, whereas RDT lack explicit mechanisms for enforcing such dual-arm coherence, leading to a lower performance. In contrast, LCM provides the necessary consistency constraints with OptimusVLA, enforcing smooth, coordinated trajectories.

4.2. Evaluation on Real-World

Training and Evaluation Setting. To validate the effectiveness of the proposed OptimusVLA in real-world robotic manipulation, we evaluate *Generalization Tasks* and *Long-horizon Tasks* on GALAXEA R1 Lite robot. It is a 14-DoF bimanual platform, and we use both wrist-mounted and third-person cameras with an image resolution of 224×224. For the *Generalization Tasks*, we train on 100–150 expert demonstrations per task and evaluate each task with 50 rollouts (25 with varying lighting conditions, and 25 with scene variations). For the *Long-horizon Tasks*, we use 200–300

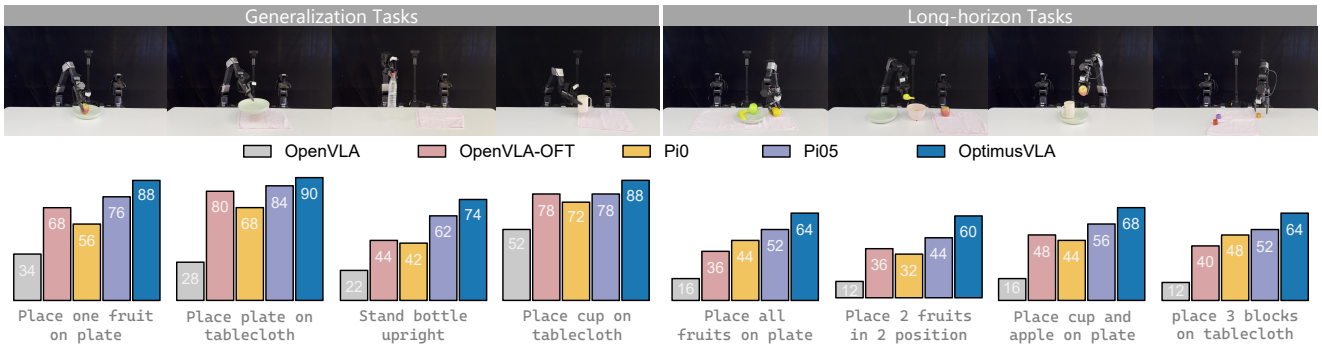
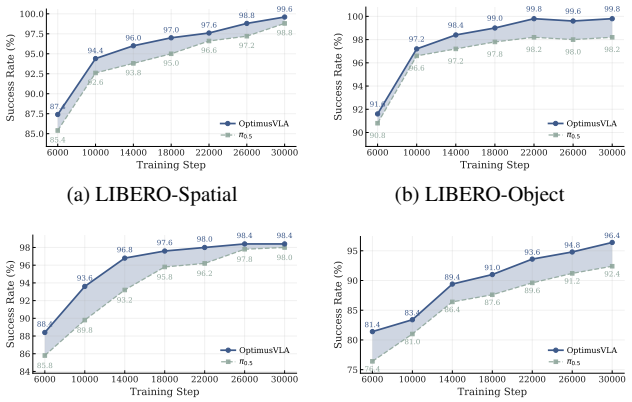


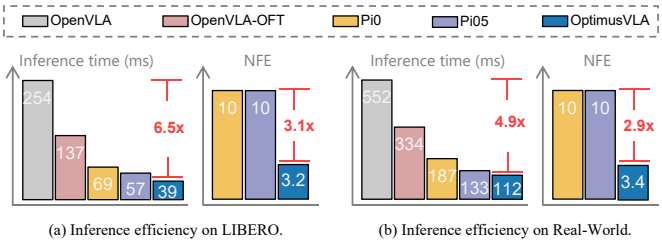
Figure 3. Real-world task setup and evaluation results. We evaluate the performance of OptimusVLA against OpenVLA [14], OpenVLA-OFT [15], π_0 [1], and $\pi_{0.5}$ [12] on the *Generalization Tasks* and *Long-horizon Tasks* suites.



(a) LIBERO-Spatial (b) LIBERO-Object

(c) LIBERO-Goal (d) LIBERO-Long

Figure 4. Training efficiency comparison between OptimusVLA and $\pi_{0.5}$ [12]. Initialized from same weights, OptimusVLA attains strong performance with substantially fewer training steps.



(a) Inference efficiency on LIBERO. (b) Inference efficiency on Real-World.

Figure 5. Inference efficiency comparison on LIBERO and Real-World. OptimusVLA attains strong performance with substantially fewer inference time and NFEs.

expert demonstrations per task for training and conduct 25 rollouts per task for evaluation. In each rollout, the objects are randomly initialized at different positions. More details are provided in **Appendix**.

Results on Real-world. As shown in Fig. 3, OptimusVLA achieves average success rates of 85.0% and 64.0% on the *Generalization Tasks* and *Long-horizon Tasks*, respectively. In particular, on the *Generalization Tasks*, OptimusVLA exhibits strong robustness to variations in lighting and scenes. We attribute this to GPM, which retrieves task-level priors conditioned on multimodal semantics, thereby rendering the policy robust to visual distractions. On *Long-horizon Tasks* that require bimanual manipulation of multiple objects, OptimusVLA outperforms π_0 by 52.4%. This showcases the superior long-horizon action stability and coherent bimanual coordination of OptimusVLA.

4.3. Ablation Study

To elucidate the benefits brought by the proposed GPM and LCM to OptimusVLA, we conduct extensive ablation studies on simulation benchmark and Real-World.

Ablation on GPM and LCM. As shown in Table 4, removing GPM causes a catastrophic performance drop of 3.8% on CALVIN and 9.4% on the *Generalization Tasks*. It can be attributed to the fact that, without GPM, OptimusVLA collapses into a standard flow-based policy whose generalization is impaired by the large gap between the prior and target distributions. This highlights the crucial role of GPM in enhancing the cross-environment and cross-scene generalization of OptimusVLA. Similarly, removing LCM causes a 1.7% drop on LIBERO-Long. This occurs because OptimusVLA loses the ability to generate temporally consistent actions. It highlights the critical role of LCM in maintaining trajectory smoothness and progress awareness.

Ablation on Memory size. As shown in Table 5, we conduct experiments to investigate how the size of the memory bank in GPM affect overall performance. We observe that performance on LIBERO-Long scales with the richness of the memory bank. Storing only one trajectory per task causes performance to degrade, as the prior becomes too deterministic. Moreover, retrieving a sufficiently large k (e.g., $k = 8$) is crucial. A small k overfits to a single retrieved trajectory, while a larger k allows GPM to construct a robust Gaussian mixture prior, balancing specificity with exploratory potential.

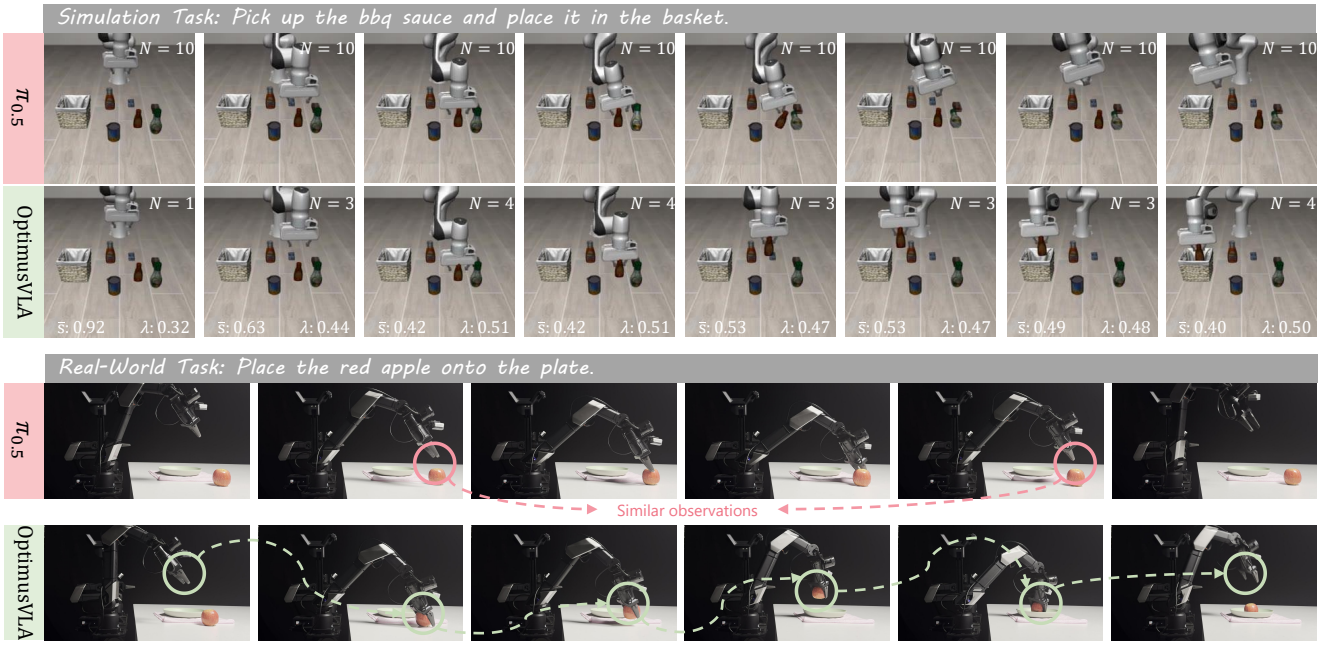


Figure 6. Qualitative results of OptimusVLA on simulation task and Real-World task. **Top**: On simulation task, we visualize the retrieval similarity $\bar{s}$, adaptive noise scales λ, and NFEs N of key frames. **Bottom**: On Real-World task, we demonstrate the role of LCM in modeling temporal dependencies, whereas $\pi_{0.5}$ struggles to distinguish between similar observations.

4.4. Efficiency Analysis

Training Efficiency. As illustrated in Fig. 4, we compare the performance of OptimusVLA and $\pi_{0.5}$ on LIBERO under different numbers of training steps. The results show that OptimusVLA achieves an 97.6% success rate on LIBERO-Goal in just 18,000 steps, while $\pi_{0.5}$ requires 26,000 steps to reach a similar level. We attribute this acceleration to the task-level priors provided by GPM. Unlike standard approaches that map from uninformative isotropic noise to the action space, OptimusVLA initiates inference from retrieved priors. This strategy effectively places the initialization in the vicinity of the target manifold, thereby substantially reducing the complexity of the transformation.
Inference Efficiency. As illustrated in Fig. 5, we compare the inference time and NFEs of OptimusVLA against $\pi_{0.5}$ on simulation benchmark and Real-World. OptimusVLA achieves 6.5× faster inference time and 3.1× fewer NFEs on LIBERO, while simultaneously delivering the highest task performance. We attribute these gains to the lightweight design of GPM and LCM, which introduce only minimal overhead yet substantially reduce the required NFEs—one of the primary factors governing inference speed.

4.5. Qualitative Analysis

In Fig. 6, we present qualitative examples from both simulation and real-world tasks. The top panel shows key frames of LIBERO task, *Pick up the bbq sauce and place it in the basket*. In $\pi_{0.5}$, the prior distribution is Gaussian noise, and actions are sampled using a fixed NFEs. This introduces a potential risk that the sampled action is infeasible. In contrast, GPM enables OptimusVLA to start inference in the vicinity of the target action distribution, with adaptive noise scale and NFEs. This results in faster inference and improved performance. The bottom panel illustrates real-world case, *Place the red apple onto the plate*. The $\pi_{0.5}$ lacks temporal awareness, thus, it struggles to determine if the apple has already been correctly placed when faced with visually similar observations. In contrast, LCM imposes a consistency constraint on OptimusVLA, leading to smoother generated trajectories and mitigating the adverse effects of similar observations.

5. Conclusion

In this paper, we propose a dual-memory VLA framework, OptimusVLA, which contain Global Prior Memory (GPM) and Local Consistency Memory (LCM) for robotic manipulation. GPM replaces Gaussian noise with task-level priors retrieved from semantically similar trajectories, thereby shortening the generative path and reducing invalid samples without sacrificing generalization. LCM models short histories of executed actions to infer task progress and injects a learned consistency constraint that enforces temporal coherence. Together, GPM and LCM improve efficiency and robustness of OptimusVLA. Extensive experiments in both simulation platforms and the real world demonstrate the su-

perior performance of OptimusVLA, together with substantially higher inference efficiency.

References

[1] 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. 1, 2, 3, 5, 6, 7

[2] Qingwen Bu, Hongyang Li, Li Chen, Jisong Cai, Jia Zeng, Heming Cui, Maoqing Yao, and Yu Qiao. Towards synergistic, generalized, and efficient dual-system for robotic manipulation. *arXiv preprint arXiv:2410.08001*, 2024. 5

[3] Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025. 3, 5

[4] Gongwei Chen, Leyang Shen, Rui Shao, Xiang Deng, and Liqiang Nie. Lion: Empowering multimodal large language model with dual-level visual knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26540–26550, 2024. 1, 4

[5] Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025. 1, 2, 5, 6, 3

[6] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44 (10-11):1684–1704, 2025. 1, 2, 5, 6, 3

[7] Shengliang Deng, Mi Yan, Songlin Wei, Haixin Ma, Yuxin Yang, Jiayi Chen, Zhiqi Zhang, Taoyu Yang, Xuheng Zhang, Wenhao Zhang, et al. Graspvla: a grasping foundation model pre-trained on billion-scale synthetic action data. *arXiv preprint arXiv:2505.03233*, 2025. 2

[8] Cunxin Fan, Xiaosong Jia, Yihang Sun, Yixiao Wang, Jianglan Wei, Ziyang Gong, Xiangyu Zhao, Masayoshi Tomizuka, Xue Yang, Junchi Yan, et al. Interleave-vla: Enhancing robot manipulation with interleaved image-text instructions. *arXiv preprint arXiv:2505.02152*, 2025. 2, 3

[9] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First conference on language modeling*, 2024. 5

[10] Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024. 5

[11] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023. 2

[12] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. pi05: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. 1, 2, 3, 5, 6, 7

[13] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022. 1, 2

[14] 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. 1, 2, 3, 5, 7

[15] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025. 2, 5, 6, 7

[16] Hao Li, Qi Lv, Rui Shao, Xiang Deng, Yinchuan Li, Jianye Hao, and Liqiang Nie. Star: Learning diverse robot skill abstractions through rotation-augmented vector quantization. In *International Conference on Machine Learning*, 2025. 1

[17] 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. 2, 5

[18] Wei Li, Renshan Zhang, Rui Shao, Jie He, and Liqiang Nie. Cogvla: Cognition-aligned vision-language-action model via instruction-driven routing & sparsification. *arXiv preprint arXiv:2508.21046*, 2025. 1, 2

[19] Wei Li, Renshan Zhang, Rui Shao, Zhijian Fang, Kaiwen Zhou, Zhuotao Tian, and Liqiang Nie. Semanticvla: Semantic-aligned sparsification and enhancement for efficient robotic manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026. 1

[20] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. *arXiv preprint arXiv:2311.01378*, 2023. 1, 2

[21] Zaijing Li, Fengxiao Tang, Ming Zhao, and Yusen Zhu. EmoCaps: Emotion capsule based model for conversational emotion recognition. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1610–1618. Association for Computational Linguistics, 2022. 3

[22] Zaijing Li, Yuquan Xie, Rui Shao, Gongwei Chen, Dongmei Jiang, and Liqiang Nie. Optimus-1: Hybrid multimodal memory empowered agents excel in long-horizon tasks. *arXiv preprint arXiv:2408.03615*, 2024. 2

[23] Zaijing Li, Yuquan Xie, Rui Shao, Gongwei Chen, Weili Guan, Dongmei Jiang, Yaowei Wang, and Liqiang Nie. Optimus-3: Dual-router aligned mixture-of-experts agent with dual-granularity reasoning-aware policy optimization. *arXiv preprint arXiv:2506.10357*, 2025. 2

[24] Zaijing Li, Yuquan Xie, Rui Shao, Gongwei Chen, Dongmei Jiang, and Liqiang Nie. Optimus-2: Multimodal minecraft agent with goal-observation-action conditioned policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9039–9049, 2025. 2, 3

[25] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 3, 4

[26] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023. 1, 2, 5

[27] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024. 1, 4

[28] Huaping Liu, Xinghang Li, Peiyan Li, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Tao Kong, and Hanbo Zhang. Towards generalist robot policies: What matters in building vision-language-action models. 2025. 5

[29] Huaping Liu, Xinghang Li, Peiyan Li, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Tao Kong, and Hanbo Zhang. Towards generalist robot policies: What matters in building vision-language-action models. 2025. 2, 3

[30] Jiaming Liu, Hao Chen, Pengju An, Zhuoyang Liu, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, et al. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025. 2

[31] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024. 6, 3

[32] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 2, 3, 4

[33] Xiao Ma, Sumit Patidar, Iain Haughton, and Stephen James. Hierarchical diffusion policy for kinematics-aware multi-task robotic manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18081–18090, 2024. 2

[34] Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3): 7327–7334, 2022. 1, 2, 5

[35] Valerio Ortenzi, Naresh Marturi, Michael Mistry, Jeffrey Kuo, and Rustam Stolkin. Vision-based framework to estimate robot configuration and kinematic constraints. *IEEE/ASME Transactions on Mechatronics*, 23(5):2402–2412, 2018. 2

[36] 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. 2, 5

[37] 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. 1, 2, 5

[38] 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. 2, 3, 4, 5

[39] Mingchen Song, Xiang Deng, Jie Wei, Dongmei Jiang, Liqiang Nie, and Weili Guan. Energyaction: Unimanual to bimanual composition with energy-based models. *arXiv preprint arXiv:2603.20236*, 2026. 1

[40] Wenxuan Song, Jiayi Chen, Pengxiang Ding, Han Zhao, Wei Zhao, Zhide Zhong, Zongyuan Ge, Jun Ma, and Haoang Li. Accelerating vision-language-action model integrated with action chunking via parallel decoding. *arXiv preprint arXiv:2503.02310*, 2025. 1, 2

[41] Wenxuan Song, Ziyang Zhou, Han Zhao, Jiayi Chen, Pengxiang Ding, Haodong Yan, Yuxin Huang, Feilong Tang, Donglin Wang, and Haoang Li. Reconvla: Reconstructive vision-language-action model as effective robot perceiver. *arXiv preprint arXiv:2508.10333*, 2025. 5

[42] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024. 5

[43] Yang Tian, Sizhe Yang, Jia Zeng, Ping Wang, Dahua Lin, Hao Dong, and Jiangmiao Pang. Predictive inverse dynamics models are scalable learners for robotic manipulation. *arXiv preprint arXiv:2412.15109*, 2024. 5

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. page 6000–6010, 2017. 5

[45] Hongyu Wang, Chuyan Xiong, Ruiping Wang, and Xilin Chen. Bitvla: 1-bit vision-language-action models for robotics manipulation. *arXiv preprint arXiv:2506.07530*, 2025. 2

[46] Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, et al. Vla-adapter: An effective paradigm for tiny-scale vision-language-action model. *arXiv preprint arXiv:2509.09372*, 2025. 1

[47] 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. 2

[48] Qianlong Xiang, Miao Zhang, Haoyu Zhang, Kun Wang, Junhui Hou, and Liqiang Nie. Tina: Text-free inversion attack for unlearned text-to-image diffusion models. *arXiv preprint arXiv:2603.17828*, 2026. 2

[49] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Gen-

eralizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024. 6, 3

[50] Haoyu Zhang, Meng Liu, Yuhong Li, Ming Yan, Zan Gao, Xiaojun Chang, and Liqiang Nie. Attribute-guided collaborative learning for partial person re-identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(12):14144–14160, 2023. 3

[51] Haoyu Zhang, Meng Liu, Zixin Liu, Xuemeng Song, Yaowei Wang, and Liqiang Nie. Multi-factor adaptive vision selection for egocentric video question answering. In *Forty-first International Conference on Machine Learning*, pages 59310–59328, 2024.

[52] Haoyu Zhang, Meng Liu, Zaijing Li, Haokun Wen, Weili Guan, Yaowei Wang, and Liqiang Nie. Spatial understanding from videos: Structured prompts meet simulation data. In *Advances in Neural Information Processing Systems*, pages 1–16, 2025. 3

[53] Jianke Zhang, Yanjiang Guo, Yucheng Hu, Xiaoyu Chen, Xiang Zhu, and Jianyu Chen. Up-vla: A unified understanding and prediction model for embodied agent. *arXiv preprint arXiv:2501.18867*, 2025. 5

[54] Qinglun Zhang, Zhen Liu, Haoqiang Fan, Guanghui Liu, Bing Zeng, and Shuaicheng Liu. Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 14754–14762, 2025. 1, 2, 4

[55] Renshan Zhang, Rui Shao, Gongwei Chen, Miao Zhang, Kaiwen Zhou, Weili Guan, and Liqiang Nie. Falcon: Resolving visual redundancy and fragmentation in high-resolution multimodal large language models via visual registers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23530–23540, 2025. 2

[56] Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, Fan Lu, He Wang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*, 2025. 1

[57] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023. 6, 3

[58] 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. *arXiv preprint arXiv:2412.10345*, 2024. 3

[59] Xurui Zhou, Gongwei Chen, Yuquan Xie, Zaijing Li, Kaiwen Zhou, Shuai Wang, Shuo Yang, Zhuotao Tian, and Rui Shao. Hiconagent: History context-aware policy optimization for gui agents. *arXiv preprint arXiv:2512.01763*, 2025. 2

[60] Yijie Zhu, Rui Shao, Ziyang Liu, Jie He, Jizhihui Liu, Jiuru Wang, and Zitong Yu. H-gar: A hierarchical interaction framework via goal-driven observation-action refinement for robotic manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026. 1

[61] 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*, pages 2165–2183. PMLR, 2023. 2

Global Prior Meets Local Consistency: Dual-Memory Augmented Vision-Language-Action Model for Efficient Robotic Manipulation

Supplementary Material

The supplementary document is organized as follows:

- Sec. A: Limitation and Future Work.
- Sec. B: Global Prior Memory.
- Sec. C: Training Details.
- Sec. D: Evaluation.
- Sec. E: Case Study.

A. Limitation and Future Work

Although OptimusVLA improves both efficiency and robustness of VLA model through Global Prior Memory and Local Consistency Memory, several limitations remain. First, the effectiveness of GPM is inherently constrained by the coverage and quality of the trajectory memory bank. When the current task or scene substantially departs from stored experiences, the retrieved priors may be misleading, potentially biasing the VLA model toward suboptimal behaviors. Second, LCM focuses on local, action-centric temporal coherence and operates on fixed-length chunks. While this design keeps the overhead small, it may be insufficient for tasks that require reasoning over very long horizons, multi-stage dependencies, or delayed effects.

A natural direction is to develop adaptive memory mechanisms, where the GPM bank is updated online with consolidation, forgetting, and uncertainty-aware retrieval, enabling continual learning and more robust behavior under distribution shift. Another avenue is to jointly train GPM, LCM, and the flow policy end-to-end. These are remains as future work for this paper.

B. Global Prior Memory

B.1. Preliminaries

We begin from Conditional Flow Matching (CFM) for action generation. Let $x_0 \sim \mathcal{P}_0$ be a source sample (typically noise) and $x_1 \sim \mathcal{P}_1$ an action from the data distribution. CFM defines a straight-line probability path between distributions:

$$x_t = (1-t)x_0 + tx_1, \qquad t \in [0,1], \tag{20}$$

implying a constant target velocity field $u_t(x_t \mid x_0, x_1) = x_1 - x_0$. The flow policy $v_\theta(t,x)$ is trained to regress this vector field by minimizing:

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, x_0\sim\mathcal{P}_0,\, x_1\sim\mathcal{P}_1} \left\| v_\theta(t,x_t) - (x_1 - x_0) \right\|_2^2. \tag{21}$$

At inference, actions are generated by solving the Ordinary Differential Equation (ODE) $dx_t/dt = v_\theta(t,x_t)$. Standard approaches set the source distribution $\mathcal{P}_0 = \mathcal{N}(0,I)$ during both training and inference. However, mapping isotropic Gaussian noise to the complex manifold of robotic actions requires a high Number of Function Evaluations (NFE). In OptimusVLA, we maintain $\mathcal{P}_0 = \mathcal{N}(0,I)$ during the pre-training of the flow policy to learn a generalizable vector field. Crucially, at inference time, GPM intervenes in this initialization stage. We construct a task-level prior $\mathcal{P}_{\mathrm{re}}$ from retrieved memories and replace the standard noise source with $\mathcal{P}_{\mathrm{re}}$. This places the starting point of the flow ODE significantly closer to the target data manifold, enabling efficient generation with reduced NFE.

B.2. Offline Construction

Let $\mathcal{D} = \{(O^{(m)}, \ell^{(m)}, J^{(m)})\}_{m=1}^{M}$ be a dataset of episodes, where $J^{(m)} = \{a_0^{(m)}, \dots, a_{T_m-1}^{(m)}\}$ is the dense action sequence. We employ the pre-trained VLM backbone to extract a compact representation for each episode. Specifically, we perform a forward pass to obtain multimodal prefix tokens $H^{(m)} \in \mathbb{R}^{L\times D_{\mathrm{vlm}}}$ and compute a mean-pooled embedding $e^{(m)}$:

$$e^{(m)} = \frac{1}{L}\sum_{j=1}^{L} H_j^{(m)} \in \mathbb{R}^{D_{\mathrm{vlm}}}. \tag{22}$$

This embedding is projected by the Prior Head f_{head} (a 2-layer MLP) into a normalized task embedding:

$$z_m = f_{\mathrm{head}}(e^{(m)}), \qquad \tilde{z}_m = \frac{z_m}{\|z_m\|_2} \in \mathbb{R}^{D_z}. \tag{23}$$

The Prior Head is trained via an InfoNCE objective to ensure that embeddings from the same semantic task are clustered together while distinct tasks are separated. Formally, for a query z_i, the loss is:

$$\mathcal{L}_{\mathrm{InfoNCE}} = -\log \frac{\sum_{j\in\mathcal{P}(i), j\neq i} \exp(\langle \tilde{z}_i, \tilde{z}_j\rangle/\tau_c)}{\sum_{k\in\mathcal{A}(i), k\neq i} \exp(\langle \tilde{z}_i, \tilde{z}_k\rangle/\tau_c)}, \tag{24}$$

where $\mathcal{P}(i)$ is the set of positive indices (same task) and $\mathcal{A}(i)$ includes all indices in the batch. After training, we freeze f_{head} and construct a Key-Value Memory Bank:

$$\mathcal{M} = \left\{(\tilde{z}_m, J^{(m)}, \mathrm{meta}^{(m)})\right\}_{m=1}^{M}, \tag{25}$$

where meta$^{(m)}$ contains trajectory metadata (length, chunking parameters). For efficient retrieval, we build a FAISS `IndexFlatIP` index over the keys $\{\tilde{z}_m\}$.

B.3. Online Retrieval and Prior Construction

At inference time t, given observation O_t and instruction ℓ, the VLM outputs current tokens H_t. We compute the query token $\tilde{z}_{re}$ similarly:

$$e_t = \frac{1}{L}\sum_{j=1}^{L} H_{t,j}, \qquad \tilde{z}_{re} = \frac{f_{\text{head}}(e_t)}{\|f_{\text{head}}(e_t)\|_2}. \tag{26}$$

Querying the memory bank yields the top-k nearest trajectories with cosine similarity scores $s_i = \langle \tilde{z}_{re}, \tilde{z}_i \rangle$. We compute soft weights α_i and a global similarity metric $\bar{s}$:

$$\alpha_i = \frac{\exp(s_i/\tau_s)}{\sum_{j=1}^{k}\exp(s_j/\tau_s)}, \qquad \bar{s} = \sum_{i=1}^{k}\alpha_i s_i. \tag{27}$$

Here, $\bar{s} \in [-1, 1]$ serves as a confidence indicator for the adaptive scheduler.

To construct the action prior, we align the retrieved full trajectories J_i to the current execution progress. We maintain a progress scalar $\rho_t \in [0, 1]$ and extract the relevant action chunk C_i from each neighbor J_i. Using sliding windows of length H_0 and stride Δ, we identify the window index $u_i = \lfloor \rho_t \cdot (N^{(i)}_{\text{chunks}} - 1) \rfloor$ and extract:

$$C_i = \text{Resample}\big(J_i[u_i \cdot \Delta : u_i \cdot \Delta + H_0]\big) \in \mathbb{R}^{H\times A}. \tag{28}$$

If the retrieved chunk length H_0 differs from the model's horizon H, we perform linear interpolation to match dimensions. The task-level prior $\mathcal{P}_{\text{re}}$ is then modeled as a Gaussian Mixture approximated by a single Moment-Matched Gaussian $\mathcal{N}(\mu, \Sigma)$. The mean μ and diagonal covariance $\Sigma = \text{diag}(\text{Var})$ are:

$$\mu = \sum_{i=1}^{k}\alpha_i C_i, \qquad \text{Var} = \sum_{i=1}^{k}\alpha_i (C_i - \mu)^{\odot 2}, \tag{29}$$

where operations are element-wise. We enforce a minimum variance floor $\sigma^2_{\min}$ to prevent collapse.

B.4. Similarity-Adaptive Sampling

GPM dynamically adjusts the generative process based on retrieval confidence $\bar{s}$. We define monotonic mappings for the noise scale λ and discretization steps N:

$$\lambda(\bar{s}) = \lambda_{\max} - \frac{\bar{s}+1}{2}(\lambda_{\max} - \lambda_{\min}), \tag{30}$$

$$N(\bar{s}) = \text{Round}\left(N_{\min} + \left(1 - \frac{\bar{s}+1}{2}\right)(N_{\max} - N_{\min})\right). \tag{31}$$

Table 6. Hyperparameter setting for each training phase.

Hyperparameter	Stage-1	Stage-2	Stage-3
Optimizer	AdamW	AdamW	AdamW
Learning Rate	5e-5	1e-4	1e-4
Steps	30000	1000	1000
Batch Size	512	64	64
Warm Up Ratio	0.10	-	-
Ema Decay	0.999	-	-

When retrieval confidence is high ($\bar{s} \approx 1$), λ decreases (relying more on the retrieved mean) and N decreases (easier transport); conversely, for novel scenarios ($\bar{s} \approx 0$), the model gracefully falls back to higher noise and more compute steps. The initialization $\hat{\mathbf{X}}_t$ for the flow policy is sampled as:

$$\epsilon \sim \mathcal{N}(0, I), \qquad \hat{\mathbf{X}}_t = \mu + \lambda(\bar{s}) \cdot \big(\epsilon \odot \sqrt{\text{Var}}\big). \tag{32}$$

This $\hat{\mathbf{X}}_t$ replaces the standard Gaussian noise x_0 in Eq. 21.

B.5. Session-Level Caching

To minimize latency, we implement a `MemorySession` module. Retrieval is performed once at the start of an episode (or when significant task drift is detected). The top-k trajectory indices and weights are cached. At each subsequent step t, the session efficiently gathers the time-aligned chunks C_i based on ρ_t and computes Eq. 29 without requerying the FAISS index, ensuring negligible overhead.

C. Training Details

We now provide a more detailed description of the three-stage training pipeline used to obtain OptimusVLA. The hyperparameter settings are shown in Table 6.

C.1. Stage I: Hierarchical VLA Pre-training

In the first stage, we pretrain a hierarchical Vision–Language–Action model following the architecture and training protocol of $\pi_{0.5}$ [12]. At this point, neither Global Prior Memory (GPM) nor Local Consistency Memory (LCM) is attached; the goal is to obtain a strong base VLA that serves as the backbone for subsequent stages.

Given an instruction ℓ and observation O_t, the Vision–Language backbone `VLM` produces a multimodal representation $E_{emb} = \texttt{VLM}(O_t, \ell)$, which is fed into the flow policy. We extract ground-truth action chunks $\mathbf{A}^\star_t \in \mathbb{R}^{H\times A}$ from the dataset and train the flow policy p_θ using a Conditional Flow Matching (CFM) objective.

C.2. Stage II: GPM Training

In the second stage, we attach a lightweight Prior Head on top of the pretrained Vision–Language backbone. All pa-

Table 7. Performance comparison on RoboTwin 2.0 [5]. We report per-task success rates (SR) and rank (RK) over 100 rollouts under *Hard* setting. † represents the result we reproduced.

Task	RDT [31]		ACT [57]		DP [6]		DP3 [49]		π_0 [1]		$\pi_{0.5}$†[12]		OptimusVLA	
	SR	RK	SR	RK	SR	RK	SR	RK	SR	RK	SR	RK	SR	RK
Click Alarmclock	12%	4	4%	7	5%	6	14%	3	11%	5	18%	2	31%	1
Click Bell	9%	3	3%	4	0%	5	0%	5	3%	4	28%	2	46%	1
Dump Bin Bigbin	32%	3	1%	6	0%	7	53%	1	24%	5	29%	4	35%	2
Open Laptop	32%	4	0%	6	0%	6	7%	5	46%	2	38%	3	48%	1
Place Bread Skillet	1%	3	0%	4	0%	4	0%	4	1%	4	2%	2	4%	1
Place Container Plate	17%	4	1%	5	0%	6	1%	5	45%	1	30%	3	37%	2
Press Stapler	24%	4	6%	5	0%	7	3%	6	29%	3	36%	2	45%	1
Stack Bowls Two	30%	4	0%	6	0%	6	6%	5	41%	3	49%	2	58%	1
Beat Block Hammer	37%	1	3%	5	0%	6	8%	4	21%	3	21%	3	26%	2
Lift Pot	9%	4	0%	5	0%	5	0%	5	36%	1	28%	3	31%	2
Move Playingcard Away	11%	4	0%	6	0%	6	3%	5	22%	3	25%	2	32%	1
Open Microwave	20%	5	0%	6	0%	6	22%	4	50%	1	39%	3	41%	2
Pick Diverse Bottles	0%	4	0%	4	0%	4	1%	3	6%	2	6%	2	7%	1
Turn Switch	15%	3	2%	6	1%	7	8%	5	23%	1	11%	4	16%	2
Place Object Stand	5%	3	0%	4	0%	4	0%	4	11%	2	11%	2	13%	1
Stack Blocks Two	2%	3	0%	5	0%	5	0%	5	1%	4	5%	2	11%	1
Average	16%	4	1%	6	1%	6	8%	5	23%	3	24%	2	30%	1

rameters of the base VLA model are frozen, and only the Prior Head is updated.

Training Procedure. We train the Prior Head to minimize the task-contrastive loss $\mathcal{L}_{\text{InfoNCE}}$ (Eq. 24). Crucially, to ensure the objective is computable and robust, we employ a Task-Pair Batch Sampler. Unlike standard random sampling, this sampler groups the training data by task ID and constructs mini-batches such that each batch contains at least two trajectories from the same task.

This strictly batched sampling strategy guarantees that every anchor sample has at least one positive pair within the current batch (in-batch positives), while all trajectories from other tasks serve as hard negatives. We optimize the head using AdamW with a learning rate of 1×10^{-4}, a batch size of 64, and a temperature $\tau_c = 0.07$ for 20 epochs.

C.3. Stage III: LCM Training

In the final stage, we train the LCM. The VLM backbone, flow policy, and GPM are all frozen.

Training Targets. LCM is trained to predict the residual needed to bridge the gap between the global prior and the ground truth. For the time step t, we first retrieve the top-k priors using GPM and compute the Gaussian mean μ_t (as defined in Eq. (29)). The regression target is:

$$\mathbf{B}_t^\star = \mathbf{A}_t^\star - \mu_t. \tag{33}$$

Optimization. We unroll the LCM model along the trajectory. The model takes the previous action chunk $\mathbf{A}_{t-1}$ as input. To ensure robustness during the initial inference steps (where no history exists), we employ a cold-start strategy during training: with probability p_{cold}, we mask $\mathbf{A}_{t-1}$ with zeros, forcing the model to rely solely on internal dynamics or output a neutral bias. The loss is the MSE between the predicted bias $\mathbf{B}_t$ and the target $\mathbf{B}_t^\star$.

D. Evaluation

D.1. Evaluation on RoboTwin 2.0

RoboTwin 2.0 defines a standardized bimanual manipulation benchmark built on the RoboTwin-OD object library, which contains 731 annotated objects from 147 categories, and a pre-collected dataset of over 100k expert dual-arm trajectories. The benchmark spans 50 collaborative dual-arm tasks instantiated on five distinct robot embodiments. For the main simulation protocol, each task is trained in a single-task manner on the Aloha–AgileX dual-arm platform using 50 clean expert demonstrations, and VLA models are tested with 100 rollouts under two difficulty settings: an *Easy* regime with uncluttered, clean scenes and a *Hard* regime with strong domain randomization over clutter, background textures, lighting, and tabletop height. In this paper, we randomly select 16 tasks for evaluation, and test the models with 100 rollouts under the *Hard* (domain-randomized with clutter, lighting, textures, and height variations) settings. As shown in Table 7, OptimusVLA achieves an average success rate of 30% across 16 tasks, surpassing

Table 8. Overview of *Generalization Tasks* on Real-World.

	Generalization Tasks			
Task name	place_one_fruit_on_plate	place_plate_on_tablecloth	stand_bottle_upright	place_cup_on_tablecloth
Description	Grasp a piece of fruit and place it onto the plate. The set of graspable fruits includes bananas, apples, lemons, and related varieties.	Grasp a plate and place it on the tablecloth. Both the plates and the tablecloths come in multiple styles.	Grasp a water bottle and then orient it to an upright pose. The bottles exhibit diverse shapes and appearances.	Grasp a cup and place it on the tablecloth. The cups exhibit diverse shapes and appearances.
# demonstrations	100	100	150	120
# rollouts	50	50	50	50
Performance (%)	88	90	74	88

Table 9. Overview of *Long-horizon Tasks* on Real-World.

	Long-Horizon Tasks			
Task name	place_all_fruits_on_plate	place_2_fruits_in_2_position	place_2_obj_on_plate	place_blocks_on_tablecloth
Description	Grasp three different fruits and place them onto the plate. For each rollout, a different combination of fruits is randomly sampled.	Grasp one fruit from the bowl and place it onto the tablecloth, then grasp another fruit from the plate and place it into the bowl.	Grasp a cup and an apple in sequence and place them into the plate.	Grasp three distinct blocks and place them on the tablecloth. For each rollout, a different combination of blocks is randomly sampled.
# demonstrations	200	200	250	300
# rollouts	25	25	25	25
Performance (%)	64	60	68	64

all baselines, including $\pi_{0.5}$.

D.2. Evaluation on Real-World

In the Real-World evaluation, we employ Galaxea R1 Lite as our robot platform. The Galaxea R1 Lite is a mobile, wheeled humanoid robot with a bimanual upper body, specifically designed for operation in human-centric indoor environments. Its embodiment comprises 23 degrees of freedom: two 6-DoF arms with spherical wrists and parallel two-finger grippers, a 3-DoF torso providing vertical and pitch motion for workspace extension, and a 6-DoF vector-drive omnidirectional base enabling coordinated whole-body manipulation.

As shown in Table 8 and Table 9, we construct two task suites: *Generalization Tasks* and *Long-horizon Tasks*. The *Generalization Tasks* primarily evaluate the model's ability to generalize across varying scenes, lighting conditions, and object instances, whereas the *Long-horizon Tasks* focus on assessing the stability and robustness of the model when executing long-horizon sequences. For each evaluation episode, we randomly initialize the objects at different spatial locations. The experimental results in Table 8 and Table 9 demonstrate the superior performance of OptimusVLA in real-world environments. We show more visualization examples in the next section.

E. Case Study

In this section, we present qualitative Real-World examples of OptimusVLA. Empowered by GPM and LCM, OptimusVLA exhibits superior performance on the *Generalization Tasks* (Figs. 7) as well as on the *Long-horizon Tasks* (Figs. 8). Moreover, we provide **unedited**, **real-time** video examples of OptimusVLA performing Real-World tasks in the supplementary materials (see `appendix/video`). Specifically, *Place plate on tablecloth* evaluates the generalization ability of OptimusVLA to varying lighting conditions and object positions, while *Place all fruits on plate* assesses the stability of OptimusVLA when executing long-horizon action sequences.

Figure 7. Qualitative results of OptimusVLA on Real-World. From top to bottom, we illustrate four *Generalization Tasks*: *Place one fruit on plate*, *Place plate on tablecloth*, *Stand bottle upright*, and *Place cup on tablecloth*. Here, `third` denotes third-person view images, while `wrist` denotes images captured from the robot's wrist-mounted camera.

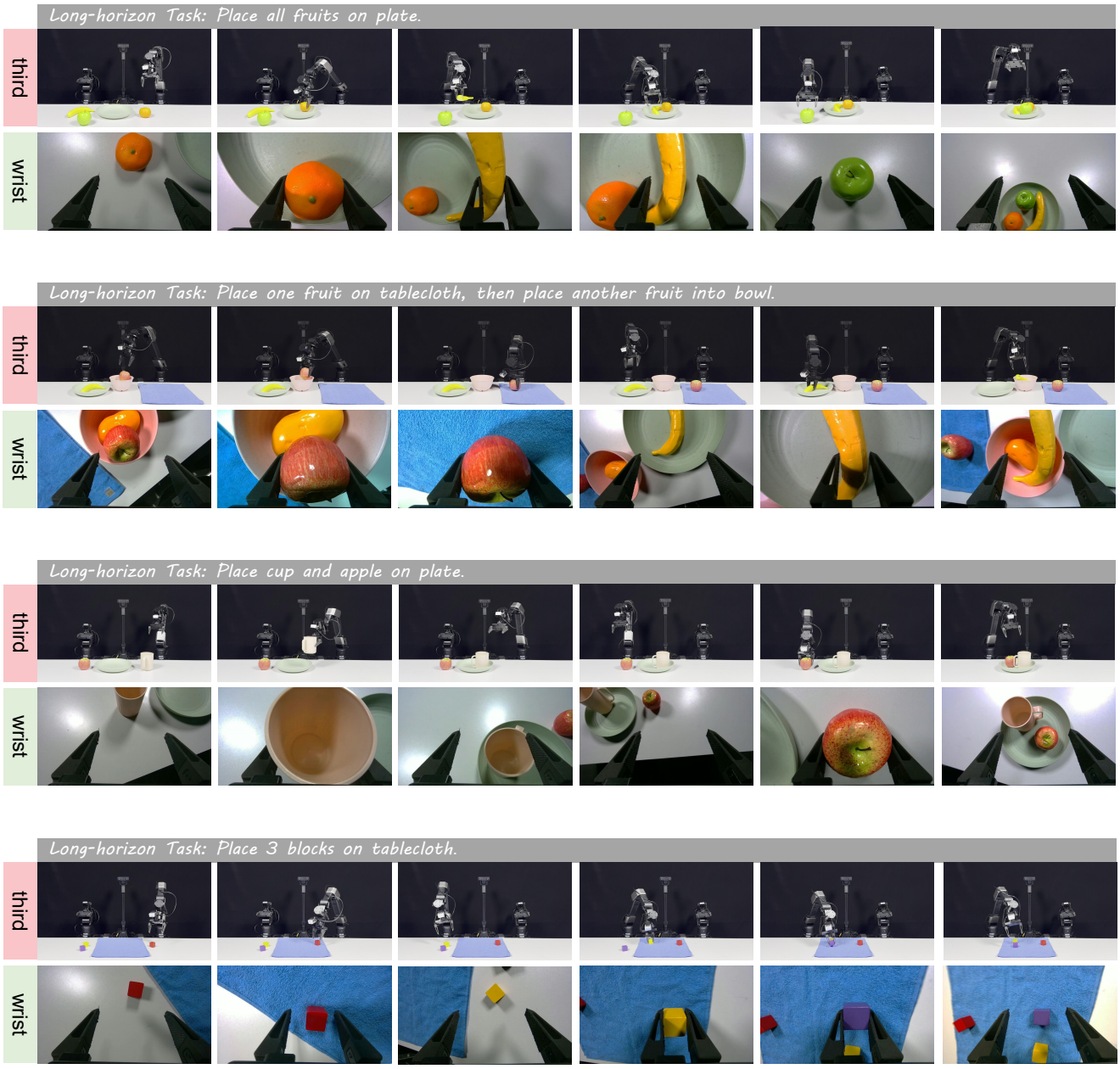


Figure 8. Qualitative results of OptimusVLA on Real-World. From top to bottom, we illustrate four *Long-horizon Tasks*: *Place all fruits on plate*, *Place one fruit on tablecloth, then place another fruit into bowl*, *Place cup and apple on plate*, and *Place 3 blocks on tablecloth*. Here, `third` denotes third-person view images, while `wrist` denotes images captured from the robot's wrist-mounted camera.