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:

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

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:

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

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 $e^{(m)}$:

$$e^{(m)} = \frac{1}{L}\sum_{j=1}^{L} H_j^{(m)} \in \mathbb{R}^{D_{\mathrm{vlm}}}. \tag{3}$$

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

$$z_m = f_{\mathrm{head}}(e^{(m)}), \qquad \tilde{z}_m = \frac{z_m}{\|z_m\|_2} \in \mathbb{R}^{D_z}. \tag{4}$$

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:

$$\mathcal{L}_{\mathrm{InfoNCE}} = -\log \frac{\sum_{j\in\mathcal{P}(i), j\neq i} \exp(\langle \tilde{z}_i, \tilde{z}_j\rangle/\tau_c)}{\sum_{k\in\mathcal{A}(i), k\neq i} \exp(\langle \tilde{z}_i, \tilde{z}_k\rangle/\tau_c)}, \tag{5}$$

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:

$$\mathcal{M} = \left\{(\tilde{z}_m, J^{(m)}, \mathrm{meta}^{(m)})\right\}_{m=1}^{M}, \tag{6}$$

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:

$$e_t = \frac{1}{L}\sum_{j=1}^{L} H_{t,j}, \qquad \tilde{z}_{re} = \frac{f_{\text{head}}(e_t)}{\|f_{\text{head}}(e_t)\|_2}. \tag{7}$$

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 $\bar{s}$:

$$\alpha_i = \frac{\exp(s_i/\tau_s)}{\sum_{j=1}^{k}\exp(s_j/\tau_s)}, \qquad \bar{s} = \sum_{i=1}^{k}\alpha_i s_i. \tag{8}$$

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:

$$C_i = \text{Resample}\big(J_i[u_i \cdot \Delta : u_i \cdot \Delta + H_0]\big) \in \mathbb{R}^{H \times A}. \tag{9}$$

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:

$$\mu = \sum_{i=1}^{k}\alpha_i C_i, \qquad \text{Var} = \sum_{i=1}^{k}\alpha_i (C_i - \mu)^{\odot 2}, \tag{10}$$

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:

$$\lambda(\bar{s}) = \lambda_{\max} - \frac{\bar{s}+1}{2}(\lambda_{\max} - \lambda_{\min}), \tag{11}$$

$$N(\bar{s}) = \text{Round}\left(N_{\min} + \left(1 - \frac{\bar{s}+1}{2}\right)(N_{\max} - N_{\min})\right). \tag{12}$$

Table 1. 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:

$$\epsilon \sim \mathcal{N}(0, I), \qquad \hat{\mathbf{X}}_t = \mu + \lambda(\bar{s}) \cdot \big(\epsilon \odot \sqrt{\text{Var}}\big). \tag{13}$$

This $\hat{\mathbf{X}}_t$ replaces the standard Gaussian noise x_0 in Eq. 2.

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. 10 without re-querying 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 1.

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 $\pi_{0.5}$ [4]. 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 2. Performance comparison on RoboTwin 2.0 [2]. We report per-task success rates (SR) and rank (RK) over 100 rollouts under *Hard* setting. † represents the result we reproduced.

Task	RDT [5]		ACT [7]		DP [3]		DP3 [6]		π_0 [1]		$\pi_{0.5}$† [4]		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. 5). 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. (10)). The regression target is:

$$\mathbf{B}_t^\star = \mathbf{A}_t^\star - \mu_t. \tag{14}$$

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 2, OptimusVLA achieves an average success rate of 30% across 16 tasks, surpassing

Table 3. 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 4. 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 3 and Table 4, 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 3 and Table 4 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. 1) as well as on the *Long-horizon Tasks* (Figs. 2). 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 1. 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.

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. 3

[2] 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. 3

[3] 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. 3

[4] Physical Intelligence, Kevin Black, Noah Brown, James

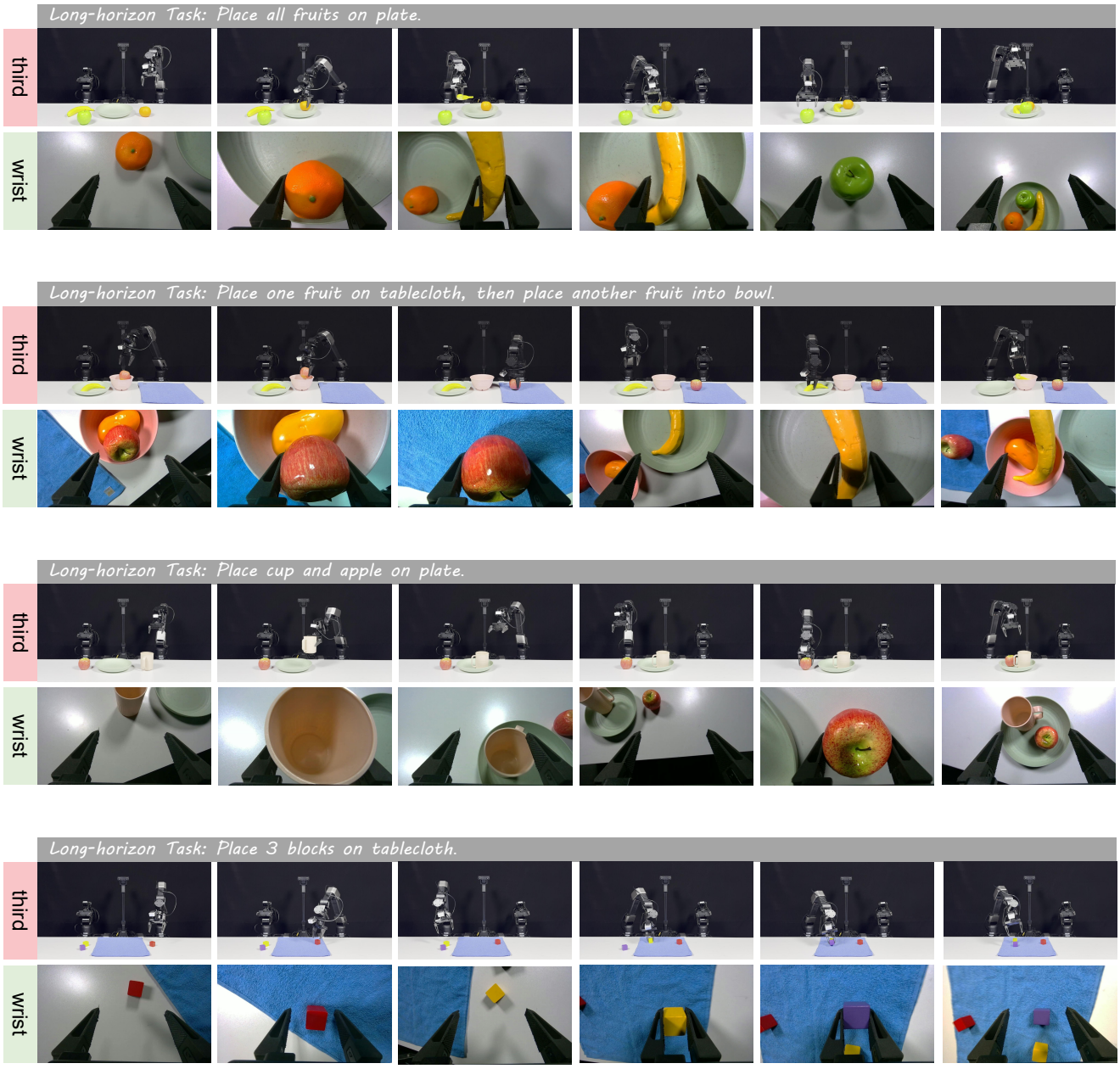

Figure 2. Qualitative results of OptimusVLA on Real-World. From top to bottom, we illustrate four *Long-horizon Tasks*: *Place all fruits on plate*, *Place one fruit on tablecloth, then place another fruit into bowl*, *Place cup and apple on plate*, and *Place 3 blocks on tablecloth*. Here, `third` denotes third-person view images, while `wrist` denotes images captured from the robot's wrist-mounted camera.

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. 2, 3

[5] 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. 3

[6] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024. 3

[7] 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. 3