ONE VIEW, ONE ACTION: SPATIALLY GENERALIZABLE VLAS THROUGH OBSERVATION-ALIGNED EXPERTS AND CROSS-GRAPH CONSISTENCY

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-view vision-language-action (VLA) models typically learn actions only in the robot's control frame, requiring prediction to account for the coordinate mapping between each camera and the robot. We propose **One View, One Action (OVOA-VLA)**: each view learns an action target expressed in its own camera coordinates. The same physical movement thus provides distinct, observation-aligned supervision for each view. We pair these targets with robot states in the corresponding frames to learn the motion required by the current robot–scene configuration, targeting spatial generalization across both camera viewpoints and robot states. OVOA-VLA jointly trains one flow-based action expert per view and a base-frame execution expert. Two complementary directed attention graphs allow the per-view experts' action losses to train the base representation and allow the base expert to use their action features. Cross-Graph Consistency encourages agreement between the two graphs' base-frame predictions for the same sample. The model supports inference with all experts or the base expert alone. Experiments on LIBERO, LIBERO-Plus, VLABench, and MimicGen show improved performance. On LIBERO-Plus, OVOA-VLA achieves 88.8% success, with gains of 9.2% and 21.2% over $\pi_{0.5}$ under camera and robot-initialization perturbations, respectively. Base-only inference retains 86.5% overall success, showing that observation-aligned supervision benefits the executable policy after removing the per-view experts. On real-world, long-horizon, out-of-distribution tasks, it outperforms $\pi_{0.5}$ by 8.3%.

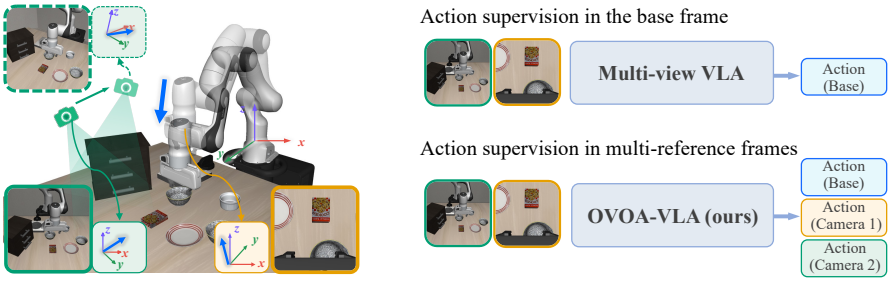
1 INTRODUCTION

The same physical robot motion can have different numerical representations in different coordinate systems (Zhang et al., 2026a). Figure 1(a) illustrates this for a single end-effector movement: its displacement components differ when expressed in the robot base coordinates and in each camera's coordinates. Yet multi-view vision-language-action (VLA) models typically aggregate observations from multiple cameras while supervising actions only in the robot base frame (Pertsch et al., 2025; Black et al., 2025a). Consequently, each view must implicitly relate the spatial information observed in its own reference frame to an action target expressed in the base frame (Zhang et al., 2026a). This cross-frame relationship varies with camera pose, making action prediction depend not only on the required physical motion but also on the mapping between the observation and control frames (Li et al., 2026b).

This motivates extending action supervision beyond the robot base frame. As shown in Figure 1(b), conventional multi-view VLAs use observations from multiple cameras to predict only a base-frame action, whereas OVOA-VLA additionally constructs action targets in the corresponding camera reference frames. Each view is therefore paired with an action representation aligned with its own reference frame, while the original base-frame supervision is retained. Jointly learning these reference-frame-specific prediction tasks allows complementary action supervision from multiple views to contribute to the executable base-frame policy.

We introduce **One View, One Action (OVOA-VLA)**, a framework for learning *observation-aligned actions*. Its organizing principle is that **each view should learn an action representation expressed**

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) The Same Action across Reference Frames (b) Comparison of Action Supervision

Figure 1: **Motivation and comparison of action supervision.** (a) The same physical action has different representations in the robot base frame and different camera frames; as the camera pose changes, its representation in the corresponding camera frame changes accordingly. (b) Existing multi-view VLAs supervise only the base-frame action, leaving the cross-frame correspondence between each view and the action implicit. We introduce additional action experts and jointly learn the base-frame action together with camera-frame actions constructed from the same demonstration, providing multi-reference-frame action supervision.

in its own reference frame. Given a multi-view demonstration, we use known camera transformations to transform the original action into the corresponding camera reference frames. Thus, the same physical motion is generally associated with different action labels across views. We construct these reference-frame-specific action targets using the known transformations and pair each image with the robot state expressed in the corresponding camera frame. A single multi-view demonstration thereby provides multiple aligned prediction tasks, each mapping the observation and robot state from one view to an action represented in that view's reference frame.

The objective is spatial generalization across both camera viewpoints and robot–scene configurations. These variations affect action prediction in different ways. A viewpoint change alters how a given movement is represented; a change in the robot's current configuration or its relationship to the scene and task goal can alter which movement is required. Reaching the same object from different end-effector poses, for example, can require different approach directions and trajectories. This dependence persists throughout execution. Our hypothesis is that predicting actions expressed in each camera frame from its image and the robot state in that frame helps this conditional relationship transfer across spatial configurations. This addresses the sensitivity to camera and robot-state changes observed in VLA evaluations (Fei et al., 2026; Zhang et al., 2026c).

Predicting actions in camera coordinates and enforcing consistency across views have been explored in prior work (Zhang et al., 2026a; Li et al., 2026b; Zhang et al., 2026b; Huang et al., 2026). Our focus is how supervision from *all available views* can jointly improve a directly executable base-frame policy. We instantiate OVOA-VLA with a Mixture of Transformers (Liang et al., 2025), comprising one action expert per input view and a base-frame expert. Each per-view expert directly attends to its corresponding visual tokens, receives robot state expressed in that camera frame, and predicts an action target expressed in the same frame. The base expert retains all visual inputs, the base-frame state, and the original executable action target.

Two complementary directed attention graphs connect these learning tasks to execution. In the base-to-reference graph, the per-view experts attend to features from the base expert, allowing their action losses to train the base representation through those attention connections. In the reference-to-base graph, the base expert attends to action features from the per-view experts, allowing its action loss to train the features it uses. The two passes share parameters and receive demonstrated-action supervision. Cross-Graph Consistency (CGC) encourages their base-frame predictions to agree for the same sample, flow timestep, and noise, while preserving each per-view expert's distinct target. Full inference uses the reference-to-base graph. The complementary graph trains the base expert without attention to the per-view experts, enabling base-only inference that retains all input views.

We evaluate OVOA-VLA on LIBERO, LIBERO-Plus, VLABench, and MimicGen. Full inference achieves 98.7% average success on LIBERO and 88.8% on LIBERO-Plus, improving over $\pi_{0.5}$ on LIBERO-Plus under camera (66.2% to 75.4%) and robot-initialization (67.2% to 88.4%) perturbations. Base-only inference retains gains under both shifts and 86.5% overall LIBERO-Plus success, showing that observation-aligned supervision benefits the executable policy after removing the per-

view experts. On real-world, long-horizon, out-of-distribution tasks with a Franka Research 3 robot, OVOA-VLA even with Base-only inference outperforms $\pi_{0.5}$ by 8.3%. Our contributions are 3-fold:

- **Observation-aligned action learning.** We introduce OVOA-VLA, which pairs each view with robot state and action targets expressed in its own camera coordinates. Jointly learning these per-view tasks and the original base-frame task turns each demonstration into multiple observation-aligned learning signals for a directly executable policy.
- **Complementary expert training.** We develop two directed attention graphs that let the per-view experts' action losses train the base representation and let the base expert use action features from the per-view experts, respectively. Cross-Graph Consistency links the graphs by regularizing their base-frame predictions for the same sample, while preserving each per-view expert's distinct target.
- **Evidence in simulation and on real robots.** We demonstrate improved manipulation performance across four simulation benchmarks and six real-world tasks, including gains under camera and robot-state perturbations in simulation. Base-only inference retains improvements after removing the per-view experts, providing evidence that per-view action supervision strengthens the executable policy through training.

2 Related Work

Vision-Language-Action Models. Recent vision-language-action (VLA) models adapt pretrained vision-language models to robotic control using large-scale robot demonstrations for direct action prediction (Pertsch et al., 2025; Kim et al., 2025b; Zitkovich et al., 2023; Black et al., 2025b). Leveraging pretrained semantic and visual knowledge, they achieve strong language-conditioned manipulation and generalization across objects, scenes, and robot embodiments (O'Neill et al., 2024; Kim et al., 2025b; Pertsch et al., 2025). VLAs mainly differ in action generation, including autoregressive discretized action tokens (Kim et al., 2025b; Zitkovich et al., 2023), diffusion-based continuous actions (Li et al., 2024), and flow matching (Black et al., 2025b).

Spatial Generalization in Vision-Language-Action Models. VLA models remain vulnerable to camera-viewpoint and robot-initialization shifts that disrupt spatial grounding (Fei et al., 2026; Zhang et al., 2026c). Existing approaches span three directions: spatial distribution augmentation diversifies training viewpoints, robot configurations, and other spatial conditions (Tian et al., 2025; Cai et al., 2026; Heo et al., 2026); geometry-aware representation learning incorporates 3D cues, geometric priors, or auxiliary spatial supervision (Li et al., 2026a; Liu et al., 2026; Li et al., 2026c); and geometry-aware action modeling uses coordinate-aware representations, action canonicalization, or geometric consistency constraints (Zhang et al., 2026a; Li et al., 2026b; Zhang et al., 2026b; Huang et al., 2026; Yuan et al., 2026). Our work follows the last direction, learning observation-aligned actions as auxiliary targets while retaining the original executable action space.

3 Method

OVOA-VLA turns observation-aligned action supervision into an executable base-frame policy (Figure 2). The central challenge is to preserve each view's own state–action mapping while transferring its learning signal to the controller. We jointly train a base execution expert and one action expert per view. Two complementary directed attention graphs allow losses from the per-view experts to update the base representation and allow the base expert to use action features from the per-view experts. Cross-Graph Consistency (CGC) regularizes the two graphs' base-frame predictions for the same sample. The graph supports deployment with all experts or with the base expert alone.

3.1 Observation-Aligned State–Action Supervision

At timestep t, a demonstration provides images $\mathcal{O}_t = \{I_t^{c_i}\}_{i=1}^{M}$ from M cameras, a language instruction l, the current robot state $s_t^b = [p_t^b, r_t^b, q_t^g]$, and an action chunk $A_t^b = [a_t^b, \ldots, a_{t+H-1}^b]$ of length H. Here b is the robot control frame; p_t^b and r_t^b encode end-effector position and orientation, and q_t^g denotes the gripper state. The base-frame target defines execution, while each view provides an additional learning task.

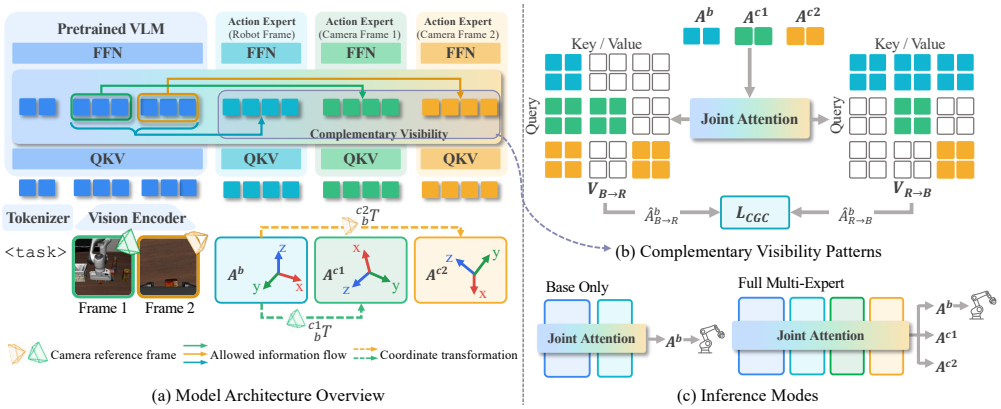


Figure 2: **Overview of OVOA-VLA.** (a) Each per-view action expert learns robot actions expressed in the corresponding camera frame and directly attends to its own view. These experts are jointly trained with a base execution expert. (b) Complementary attention graphs let the per-view experts read action features from the base expert ($B \to R$) or reverse this feature flow ($R \to B$). CGC computes the two base-frame predictions for the same sample. (c) Full inference uses $R \to B$; base-only inference omits the per-view experts while retaining all input views. Both modes execute only the base-frame prediction.

A geometric target for every view. For each input image $I_t^{c_i} \in \mathcal{O}_t$, let c_i denote the coordinate system of the camera that produced it. The known rigid transformation ${}^{c_i}_{b}T$ maps coordinates from the robot base frame b to c_i, with rotational component ${}^{c_i}_{b}R$. For wrist-mounted cameras, the transformation varies over time: the state uses the current observation timestep, while each action uses the reference timestep specified by its representation. We omit the transformation's time index for simplicity. Let e denote the end-effector frame and ${}^{b}_{e}T$ its pose encoded by s_t^b. The corresponding pose and state expressed in camera coordinate system c_i are given by ${}^{c_i}_{e}T = {}^{c_i}_{b}T\,{}^{b}_{e}T$ and $s_t^{c_i} = [p_t^{c_i}, r_t^{c_i}, q_t^g]$. The state $s_t^{c_i}$ is constructed directly from the transformed pose ${}^{c_i}_{e}T = \begin{bmatrix} {}^{c_i}_{e}R & p_t^{c_i} \\ \mathbf{0}^\top & 1 \end{bmatrix}$: $p_t^{c_i}$ is its translation component, and $r_t^{c_i}$ is obtained by converting ${}^{c_i}_{e}R$ to the same rotation representation used for r_t^b. The gripper state q_t^g remains unchanged.

The base-frame action $a_t^b = [\Delta p_t^b, \Delta r_t^b, g_t]$ consists of a translational increment, a rotational increment, and a gripper command. For the rotation-vector encoding, we first convert Δr_t^b to the rotation matrix ΔR_t^b. The translational and rotational components are then transformed as

$$\begin{cases} \Delta p_t^{c_i} = {}^{c_i}_{b}R\,\Delta p_t^b, \\ \Delta R_t^{c_i} = {}^{c_i}_{b}R\,\Delta R_t^b\,{}^{b}_{c_i}R. \end{cases} \tag{1}$$

Here ${}^{b}_{c_i}R = ({}^{c_i}_{b}R)^\top$. We then convert $\Delta R_t^{c_i}$ back to the rotation vector $\Delta r_t^{c_i}$, yielding $a_t^{c_i} = [\Delta p_t^{c_i}, \Delta r_t^{c_i}, g_t]$, while leaving the gripper command g_t unchanged. Platform-specific state and action encodings are detailed in Appendix C; Appendix B describes the coordinate conversions for these encodings. Applying the action conversion to all H steps yields $A_t^{c_i} = [a_t^{c_i}, \ldots, a_{t+H-1}^{c_i}]$. We omit time indices on the transformations above. For wrist-mounted cameras, the state uses the current observation time t; action k uses the camera pose at $t+k$ in LIBERO/MimicGen, whereas VLABench/Franka use the pose at t for the whole chunk. We jointly train the base prediction task, conditioned on $(\mathcal{O}_t, l, s_t^b)$ and supervised by A_t^b, and one task per view, conditioned on $(I_t^{c_i}, l, s_t^{c_i})$ and supervised by $A_t^{c_i}$. Each view thus receives its own numerical target for the same demonstrated movement, without additional action annotation. Conditioning on the corresponding state lets each task learn the motion required by the current robot–scene configuration throughout execution.

3.2 Observation-Aligned Flow Experts

To predict actions in the coordinate systems defined above, we adopt a Mixture of Transformers (MoT) architecture Liang et al. (2025) that retains the base action expert and introduces an independently parameterized action expert for each view. The visual-language backbone is shared, and all action experts are initialized from the pretrained $\pi_{0.5}$ action expert Black et al. (2025a).

The base expert takes the robot state s_t^b and the complete multi-view observation $\mathcal{O}_t$, and predicts the action chunk A_t^b used for robot execution. For each camera c_i, the corresponding per-view expert takes $s_t^{c_i}$, the robot state expressed in that camera's coordinates, and predicts $A_t^{c_i}$ according to the coordinate convention defined in Section 3.1. All experts are conditioned on the same language instruction l and follow the same flow-matching formulation.

To align each per-view expert's visual input with its action representation, we introduce **Reference-Aligned Visual Conditioning**. The expert associated with c_i directly attends only to visual tokens from $I_t^{c_i}$, whereas the base expert retains access to the complete multi-view observation $\mathcal{O}_t$. This design pairs each per-view expert's state and action representations with the corresponding camera observation while preserving the base expert's multi-view conditioning. Flow-matching notation and the block-independence assumptions used in the subsequent analysis are specified in Appendix B.

3.3 Complementary Attention Graphs and Cross-Graph Consistency

We use two directed graphs to connect frame-specific supervision to execution. Here B denotes the base expert and R the per-view experts, with arrows indicating feature flow. In the **base-to-reference graph** ($B \to R$), action tokens in the per-view experts attend to action tokens in the base expert. The **reference-to-base graph** ($R \to B$) reverses this attention pattern. Ordering expert blocks as $(b, c_1, \ldots, c_M)$, their visibility matrices are given by Figure 2(b).

Complementary dependencies and supervision paths. For fixed conditioning, the base flow under $B \to R$ is independent of the noisy action inputs to the per-view experts. Under $R \to B$, the base expert attends to action features from the per-view experts.

The graphs also create complementary gradient paths. Let H_b and H_{c_i} collect the expert features consumed by the receiving branches, and let θ_b, θ_{c_i} denote their private parameters. Standard back-propagation gives

$$\begin{cases} \nabla_{\theta_b} \ell_{\mathrm{FM}}^{c_i}(B \to R) = \left(\dfrac{\partial H_b}{\partial \theta_b}\right)^\top \nabla_{H_b} \ell_{\mathrm{FM}}^{c_i}, \\ \nabla_{\theta_{c_i}} \ell_{\mathrm{FM}}^{b}(R \to B) = \left(\dfrac{\partial H_{c_i}}{\partial \theta_{c_i}}\right)^\top \nabla_{H_{c_i}} \ell_{\mathrm{FM}}^{b}. \end{cases} \tag{2}$$

Thus the per-view experts' action losses can train the base representation even when the base expert does not attend to the per-view experts. Conversely, the base expert's action loss can train the features supplied by the per-view experts in the other graph. The two graphs couple the tasks in complementary ways: features from the base expert support action prediction in each camera frame, and features from the per-view experts support prediction in the execution frame.

Cross-Graph Consistency at a shared noisy base action. Evaluate both graphs using the same training sample, conditioning, τ, and the same noise samples for each expert. Let $\hat{A}^b_{\mathbf{V}}$ denote the denoised base action chunk under graph $\mathbf{V}$, obtained by subtracting the predicted base velocity scaled by τ from the noisy base action chunk. Both estimates concern the same base-frame target. For $z \sim \mathrm{Bernoulli}(0.5)$, Cross-Graph Consistency (CGC) is

$$\ell_{\mathrm{CGC}} = \begin{cases} \mathrm{MSE}(\hat{A}^b_{R\to B}, \mathrm{sg}[\hat{A}^b_{B\to R}]), & z = 0, \\ \mathrm{MSE}(\hat{A}^b_{B\to R}, \mathrm{sg}[\hat{A}^b_{R\to B}]), & z = 1. \end{cases} \tag{3}$$

Here sg preserves its argument's forward value while blocking CGC gradients through that branch. Stop-gradient applies only to the selected consistency target; both branches retain ground-truth flow supervision. Because the noisy base input is identical, CGC regularizes the base velocity's sensitivity to the interaction graph, with greater weight at larger flow timesteps. It compares predictions in the execution frame while preserving each per-view expert's own action target. Agreement is required only for the same observation and robot state. Since the $B \to R$ base branch is independent of the Gaussian noise samples $\{\epsilon^{c_i}\}_{i=1}^M$ used by the per-view experts, the expected discrepancy decomposes into the full branch's variance over these samples and its mean disagreement with the base branch. Appendix B derives this decomposition and the stop-gradient interpretation.

3.4 JOINT TRAINING AND BASE-ONLY EXECUTION

We optimize

$$\mathcal{L}_{\text{train}} = \mathbb{E}\big[\frac{1}{2} \sum_{\mathbf{V} \in \{\mathbf{V}_{B\to R}, \mathbf{V}_{R\to B}\}} (\ell^{b}_{\text{FM}}(\mathbf{V}) + \lambda_{\text{cam}} \sum_{i=1}^{M} \ell^{c_i}_{\text{FM}}(\mathbf{V})) + \lambda_{\text{CGC}} \ell_{\text{CGC}}\big], \tag{4}$$

with $\lambda_{\text{cam}} = 0.1$ and $\lambda_{\text{CGC}} = 0.02$. The expectation covers demonstrations, flow timesteps, noise, and the stop-gradient choice. Each per-view expert is supervised by action targets in its corresponding camera frame; the two graphs expose complementary routes for using this supervision, and CGC links their executable predictions.

Our model supports two deployment modes: **Full Multi-Expert Inference** and **Base Only Inference**. Both modes use the same jointly trained model without additional training, retain all input views, and use only the base-frame action for robot execution. In Full Multi-Expert Inference, we adopt the Reference-Conditioned Base Pattern ($R \to B$), allowing the base expert's action tokens to attend to action tokens from the per-view experts and thus requiring all action experts for prediction. The complementary visibility pattern ($B \to R$) used during joint training keeps the base prediction independent of the per-view experts, enabling Base Only Inference with only the base action expert while omitting the auxiliary per-view experts. Appendix B details the conditions under which this omission preserves the base prediction. The selected inference mode remains fixed throughout the rollout. Base Only Inference retains the single-action-expert inference structure, with inference cost comparable to the baseline, while Full Multi-Expert Inference uses action features from the per-view experts, providing a flexible trade-off between performance and computation.

4 SIMULATION EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Simulation Benchmarks. We evaluate our method on four simulation benchmarks to assess both spatial generalization and broader manipulation capabilities: (i) **LIBERO** (Liu et al., 2023) comprises four multi-task manipulation suites; (ii) **LIBERO-Plus** (Fei et al., 2026) extends LIBERO with controlled perturbations for evaluating out-of-distribution robustness, and we evaluate the LIBERO-trained model in a zero-shot manner; (iii) **VLABench** (Zhang et al., 2025a) is a large-scale benchmark for language-conditioned robotic manipulation, where we use Track 1 to evaluate downstream task learning and skill acquisition under in-distribution settings; and (iv) **MimicGen** (Mandlekar et al., 2023) provides manipulation tasks spanning long-horizon and high-precision behaviors, from which we select ten Panda D0 tasks to evaluate multi-task manipulation performance.

Baselines. On LIBERO and LIBERO-Plus, we compare against representative VLA methods listed in Table 1. We reproduce π_0 (Pertsch et al., 2025) and $\pi_{0.5}$ (Black et al., 2025a), while results for the remaining methods are taken from published reports, with sources cited in the table. On VLABench (Zhang et al., 2025a), we use our reproduction of $\pi_{0.5}$ as the primary baseline and include the results reported in the original benchmark paper for reference. On MimicGen, we compare our method with our reproduction of $\pi_{0.5}$ under the same experimental protocol.

Implementation Details. We initialize our model from a pretrained $\pi_{0.5}$ checkpoint and initialize all action experts with the weights of the original $\pi_{0.5}$ action expert. For all benchmarks, we train each model for 30,000 steps using a batch size of 32 on two NVIDIA RTX PRO 6000D GPUs. At inference time, the policy predicts 10-step action chunks and replans after executing the first five actions. Additional preprocessing and augmentation details are provided in Appendix D.2.

4.2 RESULTS ON LIBERO AND LIBERO-PLUS

As shown in Table 1, our model with Full Inference achieves the best performance across all four LIBERO task suites, with an average success rate of 98.7%, outperforming the strongest baseline by 1.6 percentage points. The improvement is particularly evident on the more challenging LIBERO-Long suite, where our method reaches 97.2%. On LIBERO-Plus, the full mode achieves an average success rate of 88.8% across all seven perturbation dimensions, exceeding the $\pi^{*}_{0.5}$ baseline by 10.1

Table 1: **Performance comparison on LIBERO and LIBERO-Plus.**We report success rates (%) across the four LIBERO task suites and their average. For LIBERO-Plus, we report results for the Camera and Robot perturbation dimensions, together with the average across all seven perturbation dimensions. * denotes results reproduced by us. Complete results for all seven perturbation dimensions are provided in Appendix D.3.

Model	LIBERO					LIBERO-Plus		
	Spatial	Object	Goal	Long	Avg.	Camera	Robot	Avg.
UniVLA (Bu et al., 2025)	96.5	96.8	95.6	92.0	95.2	4.3	50.3	46.4
WorldVLA (Cen et al., 2025)	85.6	89.0	82.6	59.0	79.1	0.3	30.2	24.3
π_0-FAST (Pertsch et al., 2025)	96.4	96.8	88.6	60.2	85.5	66.4	24.8	63.0
OpenVLA-OFT (Kim et al., 2025a)	97.6	98.4	97.9	94.5	97.1	59.7	37.2	72.9
DreamVLA (Zhang et al., 2025b)	97.5	94.0	89.5	89.5	92.6	65.0	40.9	69.9
MemoryVLA (Shi et al., 2026)	98.4	98.4	96.4	93.4	96.7	42.7	44.9	70.2
Libra-VLA (Wei et al., 2026)	98.6	99.4	98.0	92.8	97.2	68.9	48.8	79.5
VLA-JEPA (Sun et al., 2026)	96.2	99.6	97.2	95.8	97.2	63.3	67.1	79.5
π_0^* (Pertsch et al., 2025)	97.6	96.2	90.4	83.8	92.0	17.2	17.9	55.7
$\pi_{0.5}^*$ (Black et al., 2025a)	96.0	94.8	91.8	87.6	92.6	66.2	67.2	78.5
Ours (Base-only)	96.4	98.6	95.6	94.4	96.3	**78.4**	81.2	86.5
Ours (Full)	**99.4**	**99.8**	**98.2**	**97.2**	**98.7**	75.4	**88.4**	**88.8**

Table 2: **Results on the VLABench Track-1 in-distribution benchmark.** We report the average success rate (SR), progress score (PS), and intention score (IS) over 10 tasks. * denotes results reproduced by us. Complete results are provided in Appendix D.4.

Method	Avg. SR ↑	Avg. PS ↑	Avg. IS ↑
π_0	47.0	–	–
π_0-FAST	51.2	–	–
$\pi_{0.5}^*$	42.2	57.1	70.4
Ours (Base-only)	56.8	69.8	74.6
Ours (Full)	**60.6**	**74.2**	**77.6**

Table 3: **Results on MimicGen Panda10 under the D0 setting.** We report the average success rate (SR) over 10 tasks with 100 episodes per task. Complete results are provided in Appendix D.4.

Method	Avg. SR (%) ↑
$\pi_{0.5}^*$	64.4
Ours (Base-only)	72.7 (+8.3)
Ours (Full)	**75.3** (+10.9)

percentage points. Furthermore, it outperforms this baseline by 9.2 and 21.2 percentage points under the Camera and Robot perturbations. Notably, under Base-only Inference, our model uses only the base-frame action expert and still achieves an average success rate of 86.5% on LIBERO-Plus, with 78.4% and 81.2% under Camera and Robot perturbations. Meanwhile, its inference cost remains comparable to that of the single-action-expert baseline. Its strong robustness under Camera perturbations is consistent with the trend observed for the $B \to R$ pattern.

4.3 Evaluation across Diverse Manipulation Benchmarks

We further evaluate our method on **VLABench** (Zhang et al., 2025a) and **MimicGen** (Mandlekar et al., 2023). Compared with the common two-view input setting, VLABench provides three camera views, offering a natural setting to further examine the scalability of multi-reference action modeling with respect to the number of viewpoints. As shown in Table 2, both inference modes outperform the baseline, indicating improved manipulation precision while preserving the visual and semantic understanding required for diverse tasks. On MimicGen, both inference modes consistently outperform the $\pi_{0.5}$ baseline, as shown in Table 3. Our method achieves clear improvements on long-horizon tasks such as Coffee Preparation and high-precision tasks such as Threading, indicating that multi-reference action modeling improves complex sequential execution and fine-grained manipulation.

4.4 Ablations and Analysis

Ablations on Method Design. We use $\pi_{0.5}$ as the baseline and first add per-view action experts while retaining the base expert. At this stage, all experts share the same visual and language conditioning and independently predict actions expressed in their respective coordinate frames, without attending to one another. As shown in Table 4, adding these experts improves the success rates on LIBERO, Camera, and Robot to 96.5%, 73.1%, and 71.6%, respectively, indicating that supervision from the per-view prediction tasks benefits both in-distribution performance and robustness

Table 4: **Method design ablations on LIBERO and LIBERO-Plus.** We report success rates (%); Camera and Robot denote the corresponding perturbations in LIBERO-Plus.

Method	LIBERO Avg.	Camera	Robot
Baseline ($\pi_{0.5}$)	92.6	66.2	67.2
+ Per-View Action Experts	96.5	73.1	71.6
+ Ref.-Aligned Visual Cond.	96.4	75.7	79.6
$B \to R$ Only	96.6	**79.1**	77.0
$R \to B$ Only	98.1	70.0	85.4
Bidirectional Visibility	95.2	77.1	71.7
Complementary Visibility	98.3	74.1	86.8
+ CGC (**Ours**)	**98.7**	75.4	**88.4**

Table 5: **Performance and inference efficiency on LIBERO.** We compare the average success rate (%), computational cost per action chunk (GFLOPs), peak GPU memory usage (GB), and mean inference latency (ms) across inference modes. Lower FLOPs, memory usage, and latency indicate better efficiency.

Mode	Avg. ↑	FLOPs ↓	Mem. ↓	Lat. ↓
Baseline ($\pi_{0.5}$)	92.6	**254.65**	10.56	63.07
$R \to B$	**98.7**	263.79	19.12	103.07
$B \to R$	97.0	263.79	19.10	99.97
Base Only	96.3	254.66	**10.49**	**62.77**

to spatial perturbations. Building on this, Reference-Aligned Visual Conditioning further improves Camera and Robot success rates by 2.6 and 8.0 percentage points, respectively, while maintaining comparable performance on LIBERO. This suggests that aligning visual conditioning with the action reference frame helps improve action prediction under spatial distribution shifts.

We next compare attention patterns between the base expert and the per-view experts. The two unidirectional patterns exhibit clear complementary strengths: $B \to R$ Only reaches 79.1% under Camera perturbations, whereas $R \to B$ Only achieves 85.4% under Robot perturbations, indicating that the two interaction directions provide different advantages. In contrast, Bidirectional Visibility models both directions within a single attention graph but does not preserve the respective advantages of the two unidirectional patterns, suggesting that simply coupling both directions in one attention graph is insufficient to fully exploit this complementarity. Based on this observation, Complementary Visibility jointly trains the two unidirectional patterns to leverage information from both interaction directions, reaching 72.8% and 87.3% under Camera and Robot perturbations. Further adding Cross-Graph Consistency (CGC) brings consistent gains across all evaluation settings, improving Camera and Robot success rates by 0.7 and 1.6 percentage points, respectively, while reaching 98.7% on LIBERO. These results show that CGC further improves the joint modeling of the two visibility patterns.

Inference Efficiency. As shown in Table 5, both OVOA-VLA inference modes outperform the $\pi_{0.5}$ baseline on LIBERO. Full Multi-Expert Inference runs the per-view action experts in addition to the base expert, increasing peak memory usage and mean inference latency by approximately 81% and 61%. Nevertheless, the FLOPs remain similar across inference modes because computation is dominated by the shared visual encoder and PaliGemma prefix encoding, while the action experts process only short token sequences. Under $B \to R$, the base expert does not attend to the per-view experts. Base Only Inference therefore omits those experts, achieving an increase of **3.7** percentage points while keeping FLOPs, peak memory usage, and inference latency comparable to $\pi_{0.5}$. All efficiency metrics are measured on the same NVIDIA RTX 6000D GPU with a batch size of 1 and 10 denoising steps, with latency averaged over 100 inference runs.

5 Real-World Experiments

5.1 Experimental Setup.

We conduct real-world experiments on a 7-DoF Franka Research 3 (FR3) robot. The policy takes RGB images from a fixed third-person camera and a wrist-mounted camera, together with a task instruction, and predicts a 6-DoF end-effector (EEF) delta pose and a continuous gripper-closedness value. The evaluation includes three short-horizon tasks and three long-horizon tasks, covering basic manipulation skills and multi-stage manipulation, respectively. For each task, we collect 50 demonstration trajectories using GELLO teleoperation. During data collection, object initial positions are randomized within predefined ranges, while the third-person camera remains fixed. Demonstrations from all tasks are pooled to jointly post-train a single multi-task policy. For all real-world evaluations, we use Base Only Inference, which activates only the base-frame action expert, resulting in a single-expert inference setting comparable to that of $\pi_{0.5}$. Each method is evaluated over 20 rollouts per task, and we report task success rates. Detailed task definitions, hardware configurations, camera calibration, and training and execution settings are provided in Appendix E.2.

5.2 Real-World Manipulation Evaluation

Figure 3 presents the in-distribution results and representative rollouts across six real-world manipulation tasks. OVOA-VLA achieves an average success rate of 77.5%, surpassing $\pi_{0.5}$ at 72.5% by 5.0 percentage points. It matches the baseline on three tasks and performs better on Stack Blocks, Erase Mark, and Fill Drawer, with gains of up to 15.0 percentage points. On the long-horizon tasks, OVOA-VLA improves the average success rate from 65.0% to 73.3%, with more pronounced gains on complex, multi-stage manipulation. The right panel shows representative successful rollouts of the short-horizon Hang Mug task and the long-horizon Fill Drawer task.

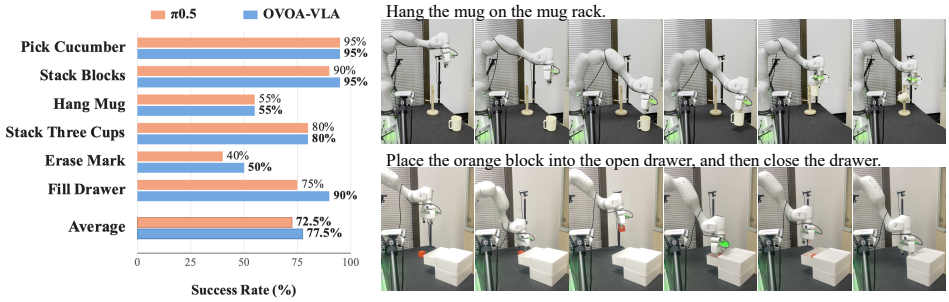


Figure 3: **Real-world manipulation evaluation.** Left: in-distribution success rates across six tasks, with 20 rollouts per task. Right: representative successful rollouts of the short-horizon Hang Mug task and the long-horizon Fill Drawer task.

5.3 Robustness under Spatial Distribution Shifts

To further evaluate real-world spatial robustness, we select three tasks and consider two types of spatial distribution shifts. For Camera Viewpoint Shift, we evaluate two unseen fixed-camera views while retaining the same wrist-mounted camera. For Robot Initialization Shift, we define two perturbed initial EEF poses while keeping all other conditions unchanged. For each task and each configuration, we conduct five real-world rollouts. All OOD experiments are evaluated zero-shot without retraining or adaptation, and detailed task selection, perturbation settings, and shift-specific results are provided in Appendix E.3. Figure 4 shows that OVOA-VLA outperforms $\pi_{0.5}$ on all three tasks, increasing the overall success rate from 28.3% to 58.3%. This consistent improvement demonstrates stronger zero-shot robustness to spatial perturbations.

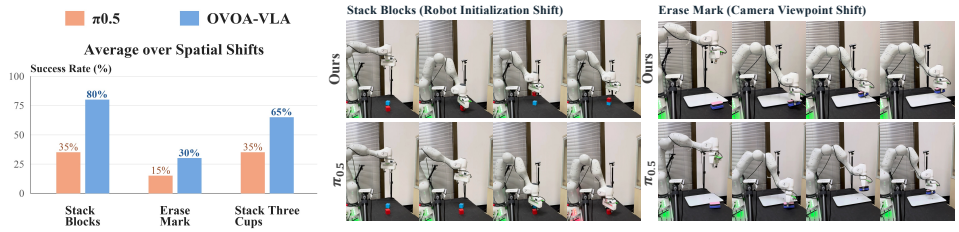


Figure 4: **Zero-shot robustness under spatial shifts.** Left: task success rates pooled across the two shift types, with 20 rollouts per task. Middle and right: representative rollouts in which OVOA-VLA succeeds and $\pi_{0.5}$ fails. Shift-specific results are shown in Figure 6.

6 Conclusion

We introduce OVOA-VLA, a framework that jointly learns observation-aligned prediction tasks using targets derived from the same demonstrations. Each per-view action expert predicts robot actions expressed in its corresponding camera frame, while the base expert predicts actions in the robot's control frame for execution. OVOA-VLA uses observation-aligned flow experts, two complementary directed attention graphs, and Cross-Graph Consistency to transfer learning signals from the per-view tasks to the base policy. Experiments demonstrate improved manipulation performance across four simulation benchmarks and six real-world tasks, with stronger robustness to camera and robot-state perturbations demonstrated in simulation. After the per-view experts are removed, base-only inference retains performance gains, with inference cost measured on LIBERO comparable to the baseline. These results show that learning an action target for each view can strengthen the executable base-frame policy during training.

7 LLM USAGE STATEMENT

Large Language Models (LLMs) were used only for minor grammar checking and language polishing. They were not involved in the research conception, methodology, experiments, analysis, or conclusions. All research content and contributions were developed solely by the authors.

REFERENCES

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, Hongyan Zhao, Hanyu Liu, Zhizhong Su, Lei Ma, Hang Su, and Jun Zhu. Motus: A unified latent action world model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 35101–35113, June 2026.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 27–30 Sep 2025a. URL https://proceedings.mlr.press/v305/black25a.html.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. pi0: A Vision-Language-Action Flow Model for General Robot Control. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025b. doi: 10.15607/RSS.2025.XXI.010.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Learning to Act Anywhere with Task-centric Latent Actions. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.014.

Boyang Cai, Qiwei Liang, Jiawei Li, Shihang Weng, Zhaoxin Zhang, Tao Lin, Xiangyu Chen, Wenjie Zhang, Jiaqi Mao, Weisheng Xu, Bin Yang, Jiaming Liang, Junhao Cai, and Renjing Xu. Beyond viewpoint generalization: What multi-view demonstrations offer and how to synthesize them for robot manipulation?, 2026. URL https://arxiv.org/abs/2603.26757.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. Worldvla: Towards autoregressive action world model, 2025. URL https://arxiv.org/abs/2506.21539.

Yu Fang, Kanchana Ranasinghe, Le Xue, Honglu Zhou, Juntao Tan, Ran Xu, Shelby Heinecke, Caiming Xiong, Silvio Savarese, Daniel Szafir, Mingyu Ding, Michael S. Ryoo, and Juan Carlos Niebles. Robotic vla benefits from joint learning with motion image diffusion, 2025. URL https://arxiv.org/abs/2512.18007.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. Libero-plus: A progressive robustness benchmark for visual-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 38574–38583, June 2026.

Hyeongjun Heo, Seungyeon Woo, Sang Min Kim, Junho Kim, Junho Lee, Yonghyeon Lee, and Young Min Kim. Anycamvla: Zero-shot camera adaptation for viewpoint robust vision-language-action models, 2026. URL https://arxiv.org/abs/2603.05868.

Bingqi Huang, Bingchuan Wei, Xuan Wang, Yingkai Cai, and Zhaokui Wang. Cross-view action consistency for camera-robust vision-language-action policies, 2026. URL https://arxiv.org/abs/2608.06965.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025a. doi: 10.15607/RSS.2025.XXI.017.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long ZENG, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=euMVC1DO4k.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation, 2024. URL https://arxiv.org/abs/2411.19650.

Wenhao Li, Xueying Jiang, Quanhao Qian, Deli Zhao, Shijian Lu, Gongjie Zhang, and Ran Xu. From fixed to free cameras: Calibration-free view-robust vision-language-action model, 2026b. URL https://arxiv.org/abs/2607.05396.

Yixuan Li, Yuhui Chen, Mingcai Zhou, and Haoran Li. Qdepth-vla: Quantized depth prediction as auxiliary supervision for vision-language-action models. In *Proceedings of the 25th International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '26, pp. 2771–2780, Richland, SC, 2026c. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9798400723179. doi: 10.65109/LJRK3716. URL https://doi.org/10.65109/LJRK3716.

Weixin Liang, LILI YU, Liang Luo, Srini Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen tau Yih, Luke Zettlemoyer, and Xi Victoria Lin. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=Nu6N69i8SB.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023. doi: 10.52202/075280-1939. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/8c3c666820ea055a77726d66fc7d447f-Paper-Datasets_and_Benchmarks.pdf.

Zhenyang Liu, Yongchong Gu, Yikai Wang, Xiangyang Xue, and Yanwei Fu. Activevla: Injecting active perception into vision-language-action models for precise 3d robotic manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8141–8151, June 2026.

Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1820–1864. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/mandlekar23a.html.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, et al. Open x-embodiment: Robotic learning datasets and rt-x models : Open x-embodiment collaboration0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903, 2024. doi: 10.1109/ICRA57147.2024.10611477.

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. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.012.

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 C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 18567–18602, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/1fa3d6ccbcd15f7285fee666b2bd57be-Paper-Conference.pdf.

Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, Guangzhong Sun, Xin Jin, and Zhibo Chen. Vla-jepa: Enhancing vision-language-action model with latent world model. In Paolo Favaro, Zuzana Kukelova, Atsuto Maki, Anna Rohrbach, Konrad Schindler, and Federico Tombari (eds.), *Computer Vision – ECCV 2026*, pp. 478–497, Cham, 2026. Springer Nature Switzerland.

Stephen Tian, Blake Wulfe, Kyle Sargent, Katherine Liu, Sergey Zakharov, Vitor Campagnolo Guizilini, and Jiajun Wu. View-invariant policy learning via zero-shot novel view synthesis. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 1173–1193. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/tian25a.html.

Yifei Wei, Linqing Zhong, Yi Liu, Yuxiang Lu, Xindong He, Maoqing Yao, and Guanghui Ren. Libra-VLA: Achieving learning equilibrium via asynchronous coarse-to-fine dual-system. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 39799–39815, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1848. URL https://aclanthology.org/2026.acl-long.1848/.

Haoqi Yuan, Zhixuan Liang, Anzhe Chen, Ye Wang, Haoyang Li, Pei Lin, Yiyang Huang, Zixing Lei, Tong Zhang, Jiazhao Zhang, Jie Zhang, Jingyang Fan, Gengze Zhou, Qihang Peng, Chenxu Lv, Xiaoyue Chen, An Yang, Fei Huang, Junyang Lin, Dayiheng Liu, Jingren Zhou, Chenfei Wu, and Xiong-Hui Chen. Qwen-robotmanip technical report: Alignment unlocks scale for robotic manipulation foundation models, 2026. URL https://arxiv.org/abs/2606.17846.

Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, and Xipeng Qiu. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11142–11152, October 2025a.

Tianyi Zhang, Haonan Duan, Haoran Hao, Yu Qiao, Jifeng Dai, and Zhi Hou. Grounding actions in camera space: Observation-centric vision-language-action policy. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(22):18782–18790, 2026a. doi: 10.1609/aaai.v40i22.38947. URL https://doi.org/10.1609/aaai.v40i22.38947.

Tianyi Zhang, Ziyang Gong, Zhenjie Yang, Zhe Qian, and Haonan Duan. Oc-vla++: Monocular geometry-guided cross-view consistency for viewpoint-robust robotic manipulation, 2026b. URL https://arxiv.org/abs/2608.01066.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, XinQiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, He Wang, Zhizheng Zhang, Li Yi, Wenjun Zeng, and Xin Jin.

Dreamvla: A vision-language-action model dreamed with comprehensive world knowledge. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 24195–24228. Curran Associates, Inc., 2025b. doi: 10.52202/085713-0818. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/22d4f952efa13970f0b1ffb22170d416-Paper-Conference.pdf.

Zhanguang Zhang, Zhiyuan Li, Behnam Rahmati, Rui Heng Yang, Yintao Ma, Amir Rasouli, Sajjad Pakdamansavoji, Yangzheng Wu, Lingfeng Zhang, Tongtong Cao, Feng Wen, Xinyu Wang, Xingyue Quan, and Yingxue Zhang. Do world action models generalize better than vlas? a robustness study, 2026c. URL https://arxiv.org/abs/2603.22078.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

A Additional Related Work

Vision-Language-Action Models Recent vision-language-action (VLA) models adapt pretrained vision-language models to robotic control by training on large-scale robot demonstrations for direct action prediction (Pertsch et al., 2025; Kim et al., 2025b; Zitkovich et al., 2023; Black et al., 2025b). Benefiting from the semantic and visual knowledge acquired during pretraining, these models have shown strong performance in language-conditioned manipulation and generalization across objects, scenes, and robot embodiments (O'Neill et al., 2024; Kim et al., 2025b; Pertsch et al., 2025). A major design choice in VLAs lies in how actions are represented and generated, ranging from autoregressive prediction of discretized action tokens (Kim et al., 2025b; Zitkovich et al., 2023) to continuous action generation with diffusion (Li et al., 2024) or flow-matching objectives (Black et al., 2025b). Despite these differences, most framework predict actions in joint or Cartesian control spaces defined independently of the observation camera, such as robot-base or world frames (Zhang et al., 2026a), leaving the geometric correspondence between camera-centered visual observations and action targets to be learned implicitly.

Mixture-of-Transformers for Multimodal Modeling Mixture-of-Transformers (MoT) employs modality-specific Transformer parameters while maintaining global self-attention over a joint token sequence, enabling specialized computation alongside cross-modal interaction (Liang et al., 2025). This expert-style separation has also been adopted in VLA models: π_0 (Black et al., 2025b) and $\pi_{0.5}$ (Black et al., 2025a) couple a VLM with a separately parameterized action expert. Motus further extends this design with visual-understanding, video-generation, and action experts in a three-expert MoT architecture (Bi et al., 2026). These designs motivate our use of MoT to model the same physical action across multiple reference frames.

B Analysis of View-Aligned Action Learning

This section derives properties of the stated objectives and attention masks. Structural statements assume the expert-block mask holds at every layer, the shared image–language prefix is independent of action tokens and robot-state inputs expressed in camera coordinates, state tokens follow their expert's visibility constraints, and normalization or routing does not introduce unmasked cross-block dependencies.

Flow-matching notation. For experts indexed by $f \in \{b, c_1, \ldots, c_M\}$, we use a shared timestep $\tau \in [0, 1]$ and independently sample $\epsilon^f \sim \mathcal{N}(0, \mathbf{I})$. The noisy action chunk is $x^f_{t,\tau} = (1-\tau)A^f_t + \tau\epsilon^f$, with target velocity $u^f_t = \epsilon^f - A^f_t$. Let $\mathcal{C}_t$ denote observation, language, and robot-state conditioning, and let $\mathbf{V}$ be the binary expert-visibility matrix, with 1 permitting attention and 0 blocking it. Expert f predicts velocity $v^f_{\theta,\mathbf{V}}$ from the noisy action inputs, τ, and $\mathcal{C}_t$ under $\mathbf{V}$. We denote the mean squared error between this prediction and u^f_t by $\ell^f_{\mathrm{FM}}(\mathbf{V})$. Independent noise means that noisy chunks are not coordinate transforms of one another; their denoising tasks are connected through the paired demonstration targets and the attention graph.

B.1 Coordinate and Temporal Conventions

The coordinate conversions in Section 3.1 act on physical poses, translational displacements, and spatial relative rotations. Rotation vectors are decoded to matrices and re-encoded after conjugation. For component-wise Euler differences, the reference and target poses must first be converted into the intended frame before their Euler-angle difference is encoded. Controller scaling and normalization must be handled consistently with these physical quantities. The implementation should be checked against the platform definitions in Appendix C.

The wrist-camera reference is part of the action label. With $u_k = t + k$ in LIBERO/MimicGen, future action labels use the camera frame at their corresponding demonstration timestep. They are not all expressed in the fixed frame of the input image at t. With $u_k = t$ in VLABench/Franka, the chunk is expressed in the observation-time camera frame. Future extrinsics in the former are training-label information only; neither inference mode needs future camera poses. The distinction must be preserved when interpreting view alignment over action chunks.

B.2 Dependencies Induced by the Expert Graphs

Fix the visual, language, and robot-state conditioning and the flow timestep. In $B \to R$, the base expert reads itself and its conditioning only. At the first layer, it therefore depends only on x^b. If this holds at one layer, the same mask preserves the dependency at the next layer. By induction, every base hidden state and its final velocity are independent of the noisy action inputs x^{c_i} to the per-view experts. Each per-view expert reads itself and the base expert, so it depends on (x^{c_i}, x^b) but not on the noisy action inputs to the other per-view experts. Reversing the edges gives the corresponding dependencies in $R \to B$.

Where derivatives exist, the velocity Jacobians with respect to the concatenated noisy actions have block structures

$$\frac{\partial \mathbf{v}_{B\to R}}{\partial \mathbf{x}} = \begin{bmatrix} J_{bb} & 0 & \cdots & 0 \\ J_{1b} & J_{11} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ J_{Mb} & 0 & \cdots & J_{MM} \end{bmatrix}, \qquad \frac{\partial \mathbf{v}_{R\to B}}{\partial \mathbf{x}} = \begin{bmatrix} K_{bb} & K_{b1} & \cdots & K_{bM} \\ 0 & K_{11} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & K_{MM} \end{bmatrix}. \tag{5}$$

These zeros follow from computational dependencies, not from learned small attention weights. In particular, the base branch in $B \to R$ is structurally independent of the Gaussian noise ϵ^{c_i} sampled for the per-view experts. The restrictions do not imply independence from the camera images, which remain available to the base expert.

Equation (2) follows by the chain rule through the features read by each receiving expert. There is no stop-gradient on those attention edges. These are available gradient paths; particular samples can still yield zero gradients. The shared visual-language backbone also receives gradients from multiple prediction tasks. Cross-graph stop-gradient in Eq. (3) does not remove the supervised cross-expert gradient routes.

B.3 What Cross-Graph Consistency Constrains

Let n be the number of action entries included in the MSE, and define $\|z\|_n^2 = \|z\|_2^2/n$. Write $Y_B = \hat{A}^b_{B\to R}$, $Y_R = \hat{A}^b_{R\to B}$, and $A = A^b_t$. All statements below concern the same observation, state, flow timestep, and shared base noisy input.

Clean-action and velocity errors. For the affine flow path, $x^b_\tau = A + \tau u^b$. Thus

$$Y_g - A = \tau(u^b - v^b_g), \qquad \|Y_R - Y_B\|_n^2 = \tau^2 \|v^b_R - v^b_B\|_n^2. \tag{6}$$

The identity concerns the same numerical action space. It need not survive nonlinear decoding into another representation. It also does not imply an equivariance constraint between per-view experts, whose targets and noise draws differ.

Random stop-gradient. Set $D = \|Y_R - Y_B\|_n^2$ and let $J_g = \partial Y_g/\partial\theta$ for the shared model parameters. With equal probability for the two stop-gradient directions, the expected gradient produced by autodifferentiation is

$$\mathbb{E}_z[\nabla_\theta \ell_{\mathrm{CGC}}] = \frac{1}{n}(J_R - J_B)^\top (Y_R - Y_B) = \frac{1}{2}\nabla_\theta D. \tag{7}$$

The loss value is D for either direction, but the gradient is routed through only one branch in each sample. In expectation this is one-half of the undetached symmetric-MSE gradient. Random routing therefore does not, by itself, provide a different optimum or a guarantee of superior distillation.

Sensitivity to auxiliary action noise. Fix a training demonstration, τ, and ϵ^b, so Y_B is fixed under the graph assumptions. Let the remaining randomness be the Gaussian noise sampled for the per-view experts, $\epsilon^R = (\epsilon^{c_1}, \ldots, \epsilon^{c_M})$, and write $\bar{Y}_R = \mathbb{E}_{\epsilon^R}[Y_R]$. Expanding the squared difference gives

$$\mathbb{E}_{\epsilon^R}[D] = \mathbb{E}_{\epsilon^R}\left[\|Y_R - \bar{Y}_R\|_n^2\right] + \|\bar{Y}_R - Y_B\|_n^2. \tag{8}$$

The cross term vanishes because $\mathbb{E}[Y_R - \bar{Y}_R] = 0$. Thus the expected discrepancy penalizes variability of the full prediction induced by the noise sampled for the per-view experts and disagreement

between its conditional mean and the closed base branch. Additional stochastic layers must also be fixed or included explicitly in the conditioning for this decomposition. Sampling noise independently for each expert does not make the noisy chunks statistically independent when their clean targets share a demonstration.

B.4 When Base-Only Inference Preserves the Base Trajectory

The base conditioning in $B \to R$ is $\mathcal{C}_t^b = (\mathcal{O}_t, l, s_t^b)$, with no information from the robot-state inputs $s_t^{c_i}$ in the shared image–language prefix. By the dependency argument above, the unpruned base velocity is $F_b(x^b, \tau, \mathcal{C}_t^b)$ and depends on neither the noisy action inputs x^{c_i} to the per-view experts nor their robot-state inputs $s_t^{c_i}$. Removing the per-view experts and their state inputs leaves this function unchanged if the base computations and conditioning are preserved. With the same initial base noise and a fixed integration grid, the first numerical update is therefore identical. Induction over integration steps gives identical subsequent base flow states x^b. The argument extends to multi-stage solvers whose stages and grid are identical between runs.

For an adaptive solver, the step-selection rule must also be independent of the omitted flow states x^{c_i} of the per-view experts; otherwise the numerical trajectories can differ even when the continuous base dynamics agree. Stochastic inference must couple any randomness affecting the base. Robot-state inputs $s_t^{c_i}$ or action features from the per-view experts relayed to the base through a shared prefix would invalidate the required independence. These conditions should be checked in the implementation. The result equates pruned and unpruned execution of $B \to R$ only. Agreement with $R \to B$ is learned through CGC and is not an exact structural property.

C Action Representation Details

Table 6 summarizes the proprioceptive states and action representations used across platforms. LIBERO and MimicGen retain the native seven-dimensional incremental control commands, comprising three translational components, three rotational components in rotation-vector form, and one gripper command, with the original controller scales preserved. For VLABench and real-world Franka, absolute target poses are encoded relative to the end-effector pose observed at the start of the action chunk. VLABench uses position differences and component-wise XYZ Euler-angle differences, whereas Franka uses position differences and a rotation-vector representation of the relative orientation. All targets within a chunk share this initial pose as their reference. Gripper components retain their platform-specific definitions; for Franka, gripper closedness ranges from 0 (fully open) to 1 (fully closed). Actions are executed in the robot base frame on all platforms. For LIBERO and MimicGen, denormalized predictions are passed directly to the environment; for VLABench and Franka, predictions are first decoded into absolute target poses and then passed to the respective execution interfaces.

When constructing wrist-frame action supervision, we adopt temporal reference conventions consistent with each platform's action definition. For LIBERO and MimicGen, each action in a chunk is transformed using the wrist-camera extrinsics at its corresponding timestep. For VLABench and real-world Franka, the entire chunk uses the wrist-camera extrinsics at the initial observation timestep, which are updated upon replanning. Future camera extrinsics are used only to construct training targets; action execution does not require future camera poses.

D Additional Details of Simulation Experiments

D.1 Benchmark and Dataset Details

LIBERO and LIBERO-Plus. The four task suites in LIBERO (Liu et al., 2023) emphasize spatial relationships, manipulated objects, task goals, and long-horizon manipulation, respectively, providing a common environment for evaluating policies across different task requirements. We use all 40 tasks from the Spatial, Object, Goal, and Long suites and jointly train a single multi-task policy on the publicly released LIBERO demonstrations, with 50 trajectories per task and 2,000 trajectories in total. The original images have a resolution of 128×128, and we do not filter out failed trajectories before training. Each task is evaluated over 50 rollouts. LIBERO-Plus (Fei et al., 2026) organizes

Table 6: **Proprioceptive state and action representations.** Dimensions exclude padding. Numbers in parentheses indicate component dimensions. EEF denotes the end effector.

Platform	Type	Dim.	Position	Orientation	Gripper
LIBERO / MimicGen	State	8	EEF position (3)	Rotation vector (3)	Finger joint positions (2)
	Action	7	Translational increment command (3)	Rotation-vector increment command (3)	Gripper command (1)
VLABench	State	7	EEF position (3)	XYZ Euler angles (3)	Gripper state (1)
	Action	7	Target position offset (3)	Component-wise XYZ Euler-angle difference (3)	Gripper command (1)
Real-world Franka	State	8	EEF position (3)	Quaternion in $wxyz$ order (4)	Gripper closedness (1)
	Action	7	Target position offset (3)	Relative rotation vector (3)	Target gripper closedness (1)

perturbations into dimensions, enabling separate assessments of policy sensitivity to different environmental changes. We evaluate all seven official perturbation dimensions, following the official evaluation configurations and rollout counts. All LIBERO-Plus evaluations are conducted zero-shot using the same LIBERO-trained model, without training or adaptation on any perturbed data.

VLABench. VLABench (Fang et al., 2025) is a benchmark for language-conditioned robotic manipulation, with multiple evaluation tracks assessing task learning and generalization. We adopt the in-distribution setting of Track 1, which evaluates the ability of policies to learn downstream tasks and acquire manipulation skills from demonstrations. We use the 10 primitive tasks in the fine-tuning setup, with 500 demonstration trajectories per task and 5,000 trajectories in total, and jointly train a single multi-task policy across all tasks. Visual observations comprise two fixed camera views and one wrist-mounted camera view, each with a resolution of 480×480. The robot state includes the end-effector position and orientation, along with the gripper state. Evaluation follows the official Track 1 task configurations, with 50 rollouts per task. We report success rate (SR), progress score (PS), and intention score (IS), averaged over the 10 tasks. SR measures overall task completion, PS measures execution progress as the proportion of completed subtasks, and IS reflects the tendency to select the correct target object based on whether the end effector approaches it. We treat SR as the primary evaluation metric, while PS and IS provide complementary measures for understanding task progress and target selection.

MimicGen. MimicGen (Mandlekar et al., 2023) is a framework for generating robotic manipulation demonstrations by transforming and adapting existing trajectories to different tasks and scenes. Its tasks include both long-horizon, multi-stage manipulation and precision-demanding assembly. For example, Coffee Preparation involves multiple stages, whereas Threading requires precise alignment and insertion. We select 10 D0 tasks using the Franka Emika Panda robot from the publicly released core dataset: Stack, Square, Coffee, Threading, Stack Three, Three Piece Assembly, Hammer Cleanup, Mug Cleanup, Coffee Preparation, and Kitchen. D0 denotes the default object initialization distribution, which we use for both training and evaluation. We jointly train a single multi-task policy using 1,000 demonstration trajectories per task, totaling 10,000 trajectories. Observations include third-person and wrist-camera images rendered at 224×224 resolution, together with the robot state. We establish a unified evaluation protocol for these tasks, with 100 rollouts per task and the same pregenerated, fixed simulator initial states across all methods. We use the environments' native success conditions and terminate each rollout upon success or when the maximum episode length is reached. The maximum episode length is 400 steps for Stack, Square, Coffee, Threading, and Stack Three; 500 steps for Three Piece Assembly, Hammer Cleanup, and Mug Cleanup; and 800 steps for Coffee Preparation and Kitchen.

D.2 Training and Implementation Details

Training Configuration. We use the AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$, a weight decay of 10^{-10}, and a global gradient norm clipping threshold of 1.0. For LIBERO and MimicGen, the learning rate is linearly warmed up to 5×10^{-5} over the first 10,000 updates and remains constant thereafter. For VLABench, the learning rate is linearly warmed up to 2.5×10^{-5} over the first 1,000 updates, followed by cosine decay to 2.5×10^{-6}. All images are resized to 224×224 before being fed into the model. During training, we apply random color jitter to images from all camera views, varying brightness, contrast, and saturation to diversify visual appearance and reduce reliance on fixed lighting and color conditions. The corresponding augmentation strengths are set to 0.3, 0.4, and 0.5, respectively. For the third-person views in LIBERO and MimicGen and the primary fixed-camera view in VLABench, we additionally apply random cropping and in-plane rotation. The crop spans 95% of the input width and height and is resized back to the original input dimensions, while the rotation angle is sampled from $[-5^\circ, 5^\circ]$. These random augmentations are enabled only during training. We train the $\pi_{0.5}$ baseline on all three datasets using the same training data, optimizer, learning-rate schedule, and image preprocessing and augmentation settings as our method.

Software and Simulation Environments. We use JAX/Flax for model training and inference, with separate software environments for each simulation benchmark. LIBERO and LIBERO-Plus both use MuJoCo 3.2.3, paired with robosuite 1.4.1 and 1.4.0, respectively. VLABench uses MuJoCo 3.2.2. The MimicGen environment uses MimicGen 1.0.1, robosuite 1.4.1, robomimic 0.3.1, and MuJoCo 2.3.2.

D.3 Complete Results on LIBERO-Plus

Table 7 presents the results across all seven perturbation dimensions. Beyond Camera and Robot, both inference modes outperform the reproduced $\pi_{0.5}$ baseline on Language, Light, Background, Noise, and Layout, showing that the improvements extend to a broader range of perturbations. Full mode achieves the best results among the compared methods on all five dimensions, while Base Only mode maintains strong performance and outperforms Full mode under Camera perturbations.

Table 7: **Performance comparison on LIBERO-Plus.** We report success rates (%) under all seven perturbation dimensions of LIBERO-Plus, together with the average performance. * denotes results reproduced by us.

Method	Camera	Robot	Language	Light	Background	Noise	Layout	Average
UniVLA	4.3	50.3	71.8	59.1	80.0	25.3	34.3	46.4
WorldVLA	0.3	30.2	44.2	29.4	14.5	12.2	39.4	24.3
π_0-FAST	66.4	24.8	63.3	73.0	67.7	75.8	70.3	63.0
OpenVLA-OFT	59.7	37.2	81.5	85.8	92.4	76.7	77.1	72.9
DreamVLA	65.0	40.9	63.5	85.7	82.7	85.0	74.0	69.9
MemoryVLA	42.7	44.9	84.4	92.8	95.0	62.1	84.7	70.2
Libra-VLA	68.9	48.8	92.7	97.9	93.4	86.3	77.5	79.5
VLA-JEPA	63.3	67.1	85.4	95.6	93.6	66.3	85.1	79.5
π_0^*	17.2	17.9	69.3	90.1	88.1	75.7	31.9	55.7
$\pi_{0.5}^*$	66.2	67.2	75.1	91.7	90.3	78.1	81.0	78.5
Ours (Base Only)	**78.4**	81.2	82.6	94.6	94.0	88.8	85.6	86.5
Ours (Full)	75.4	**88.4**	**84.8**	**97.7**	**95.9**	**90.1**	**89.4**	**88.8**

D.4 Complete Results on VLABench and MimicGen

The complete task-wise results on **VLABench** are presented in Table 9. Both inference modes outperform the reproduced $\pi_{0.5}$ baseline overall, with Full mode performing slightly better. On Select Chemistry Tube, Base Only and Full improve the success rate from 6% to 78% and 70%, respectively. Both modes also improve success rates and progress scores on Select Book, Select Mahjong, and Select Poker, while their relative strengths vary somewhat across tasks. For Select

Chemistry Tube and Select Toy, the target entities or distance criteria used by IS do not fully align with the task success conditions. Their IS values can therefore serve as supplementary references but should not be used alone to assess the policy's understanding of the target.

For **MimicGen**, we provide a breakdown of performance across the ten Panda D0 tasks in Table 8. Both inference modes achieve higher average success rates than the baseline, with Full mode performing slightly better. On the multi-stage Coffee Preparation task, Base Only and Full improve the success rate from 32% to 65% and 72%, respectively. On Threading, which requires precise alignment and insertion, both modes improve the success rate from 35% to 42%. These results demonstrate improvements on both long-horizon and precision-demanding manipulation tasks.

Table 8: **Performance comparison on MimicGen D0 distribution.** We report the success rate (%) for each task and the average performance.* denotes results reproduced by us.

Method	Coffee	Coffee Prep.	Hammer	Kitchen	Mug	Square	Stack	Stack 3	Thread	3P Assembly	Avg.
$\pi^*_{0.5}$	71	32	97	99	44	58	97	79	35	32	64.4
Ours (Base Only)	**85**	65	**100**	**100**	48	71	**100**	78	**42**	**38**	72.7
Ours (Full)	84	**72**	**100**	**100**	**56**	**78**	98	**90**	**42**	33	**75.3**

Table 9: **Task-wise performance on VLABench.** We report Success Rate (SR), Progress Score (PS), and Intention Score (IS), with all values expressed as percentages. The Average row reports the unweighted mean across the ten tasks. * denotes results reproduced by us.

Task	$\pi^*_{0.5}$			Ours (Base Only)			Ours (Full)		
	SR	PS	IS	SR	PS	IS	SR	PS	IS
Add Condiment	54	75.3	98	50	72.7	98	**66**	80.7	96
Insert Flower	22	56	86	26	56	84	**30**	63	100
Select Book	50	56	76	64	73	88	**74**	82	98
Select Chemistry Tube	6	31	2	**78**	87	2	70	85	2
Select Drink	54	66	90	58	71	92	**68**	76	92
Select Fruit	46	66	94	**50**	71	96	44	71	100
Select Mahjong	34	41	92	56	59	92	**70**	76	98
Select Painting	34	34	62	**44**	44	70	42	42	66
Select Poker	74	75.3	84	88	90	98	**92**	93.3	100
Select Toy	48	70	20	**54**	74	26	50	73	24
Average	42.2	57.1	70.4	56.8	69.8	74.6	**60.6**	**74.2**	**77.6**

D.5 Qualitative Results across Simulation Benchmarks

Figures 9, 10, 11, and 12 present successful execution trajectories of our method across four simulation benchmarks. For LIBERO, we show one trajectory from each of the four task suites; for LIBERO-Plus, one trajectory under each of the seven perturbation types. For VLABench and MimicGen, we show one successful trajectory for every evaluated task. Task names are annotated in the figures, and temporally ordered keyframes illustrate the execution process.

E Real-World Experimental Details

E.1 Task Definitions

We consider three short-horizon tasks and three long-horizon tasks, covering color and spatial understanding, pick-and-place manipulation, precise alignment, tool use, and articulated-object interaction. Table 10 summarizes the language instructions, task setups, and success criteria. For long-horizon tasks, success requires completion of all specified objectives.

Table 10: **Real-world task definitions.** We describe the language instruction, task setup, and success criterion for each short-horizon and long-horizon task.

Task	Language Instruction	Task Setup and Initialization	Success Criterion
Short-horizon tasks			
Pick Cucumber	*"Pick up the cucumber and place it into the blue bowl."*	A cucumber and two differently colored bowls are placed on the table. The robot must identify the blue bowl and place the cucumber inside it. During data collection, the cucumber's initial position is randomized, and the positions of the two bowls are randomly exchanged.	After the gripper releases the cucumber, it lies entirely inside the blue bowl.
Stack Blocks	*"Place the red block on top of the blue block."*	The robot must pick up the red block and stack it on the blue block. Both blocks' initial positions are randomized without a fixed left–right ordering, with small variations in their in-plane orientations.	After release, most of the red block's bottom face rests on the blue block, and the stack remains stable. Exact centering is not required.
Hang Mug	*"Hang the mug on the mug rack."*	The robot must grasp the mug, adjust its orientation, and align its handle with a hook on the rack. The mug's initial position and handle orientation are varied during data collection.	The handle is placed over the hook, and the mug remains stably supported by the rack after the gripper releases it.
Long-horizon tasks			
Stack Three Cups	*"Stack all the cups on the leftmost cup."*	Three equally sized cups of different colors are placed separately on the table. The robot must sequentially stack the other two cups onto the initially leftmost cup. Cup positions vary within their designated regions, with variations in relative placement and small orientation perturbations.	The initially leftmost cup remains at the bottom, and the other two cups are stacked above it, forming a single stable three-cup stack after release.
Erase Mark	*"Pick up the whiteboard eraser and erase the black X on the whiteboard."*	A whiteboard is fixed horizontally on the table, with a black X drawn on its surface and an eraser placed beside it. The robot must grasp the eraser, move it to the mark, and perform consecutive wiping motions.	The robot grasps the eraser and removes most of the black X while the whiteboard remains stable throughout the operation.
Fill Drawer	*"Place the orange block into the open drawer, and then close the drawer."*	The drawer is initially open, and an orange block is placed on the table. The robot must first place the block inside the drawer and then push the drawer closed.	The orange block is inside the drawer, and the drawer is fully closed.

E.2 Implementation Details

Hardware Setup. As shown in Figure 5, we conduct real-world experiments using a 7-DoF Franka Research 3 (FR3) robot equipped with a parallel-jaw gripper, and collect demonstrations using GELLO teleoperation. Visual observations are captured by Intel RealSense D415 cameras at an RGB resolution of 640×480 pixels and 30 FPS. Only RGB streams are used. The setup comprises one wrist-mounted camera and three fixed third-person cameras. The policy always receives two visual inputs: the wrist-mounted view and one selected third-person view. During data collection and in-distribution evaluation, we use the same reference third-person camera together with the wrist-mounted camera. For camera viewpoint shift evaluation, the third-person input is switched to one of the other two fixed cameras, while the wrist-mounted input and all other experimental conditions

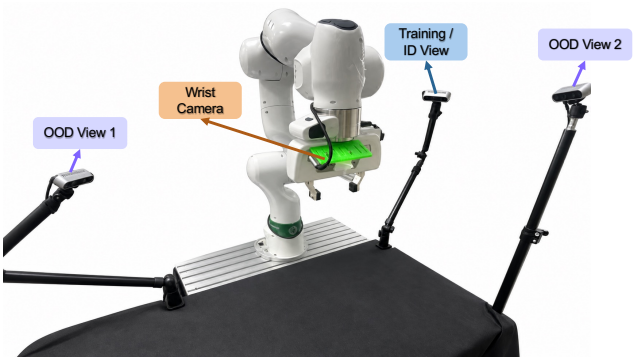


Figure 5: **Real-world setup.** The training and additional test cameras are indicated.

remain unchanged. Camera roles are annotated in the figure, and detailed viewpoint configurations are provided in Appendix E.3.

Camera Calibration. To obtain the base-to-camera transformations ${}^{c_i}_{b}T$ used in our method, we calibrate the extrinsics of two fixed cameras and one wrist-mounted camera. We use a ChArUco board and collect images together with the corresponding robot end-effector poses at multiple positions and orientations. Given the camera intrinsics and detected board corners, we estimate the board pose in each camera frame using PnP, then solve for the camera extrinsics using OpenCV hand-eye calibration. Below, m denotes the calibration board frame, and k indexes calibration samples.

For each fixed camera, the board is rigidly attached to the end effector, yielding

$$ {}^{c_i}_{m}T(k) = {}^{c_i}_{b}T \, {}^{b}_{e}T(k) \, {}^{e}_{m}T. \tag{9} $$

Here, ${}^{c_i}_{m}T(k)$ is estimated from the image, ${}^{b}_{e}T(k)$ is provided by the robot, and ${}^{e}_{m}T$ remains constant throughout calibration. We perform eye-to-hand calibration using these observations to obtain ${}^{c_i}_{b}T$ for each fixed camera.

For the wrist-mounted camera, the board is fixed in the workspace, and the poses satisfy:

$$ {}^{c_w}_{m}T(k) = {}^{c_w}_{e}T \left({}^{b}_{e}T(k) \right)^{-1} {}^{b}_{m}T, \tag{10} $$

where ${}^{b}_{m}T$ remains constant. Eye-in-hand calibration yields the fixed end-effector-to-camera transformation ${}^{c_w}_{e}T$. At each observation timestep t, we compute the base-to-wrist-camera transformation from the current end-effector pose:

$$ {}^{c_w}_{b}T(t) = {}^{c_w}_{e}T \left({}^{b}_{e}T(t) \right)^{-1}. \tag{11} $$

These extrinsics are used for the state and action transformations defined in the main text. In our real-world implementation, all actions within a predicted chunk use the wrist-camera frame at the initial observation timestep. The transformation is updated when a new observation is acquired for replanning.

Training and Execution. Our method and the reproduced $\pi_{0.5}$ baseline are trained on the same pooled multi-task demonstration dataset using the corresponding language task instructions, and share the same main training hyperparameters. Both models are trained for 30K steps with a global batch size of 64 using AdamW. The learning rate is warmed up to 2.5×10^{-5} over the first 1,000 steps, followed by cosine decay toward 2.5×10^{-6}. Demonstration trajectories are sampled at 30 Hz for training. Both policies predict 7-D Cartesian actions consisting of a 3-D translational offset, a 3-D world-relative rotation vector, and a continuous target gripper-closedness value in $[0, 1]$. At each inference step, the policy predicts a 10-step action chunk, with all actions decoded relative to the end-effector pose observed at the beginning of the chunk. The first five actions are executed before acquiring new observations and replanning. The continuous gripper prediction is converted into open/close commands using hysteresis: values at or below 0.3 trigger opening, values at or above 0.7 trigger closing, and intermediate values retain the previous gripper state.

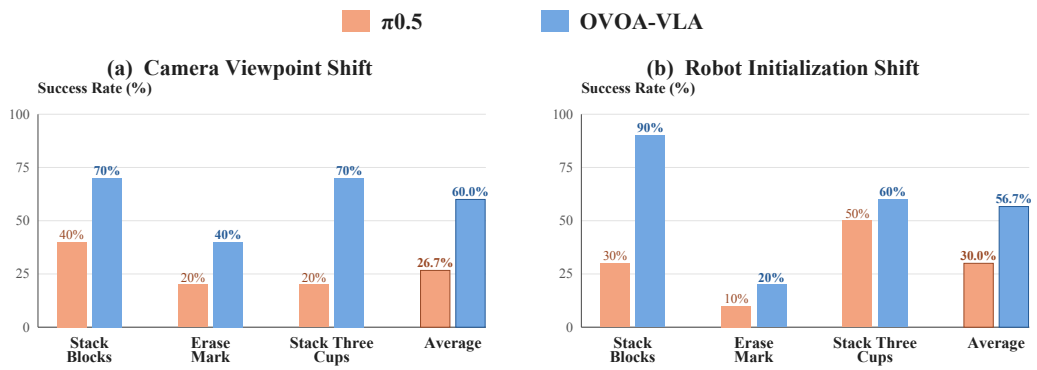


Figure 6: **Detailed zero-shot robustness under spatial distribution shifts.** Success rates under camera viewpoint and robot initialization shifts. Each task aggregates two OOD configurations with five rollouts per configuration; Average denotes the mean across the three tasks.

E.3 Spatial Distribution Shift Details

As shown in Figure 5, training and in-distribution evaluation use the same fixed third-person camera together with the wrist-mounted camera. For the camera viewpoint shift, the training third-person view is replaced with either OOD View 1 or OOD View 2, neither of which is used during demonstration collection or post-training, while the wrist-mounted view remains unchanged. For the robot initialization shift, the robot is first reset to its nominal task-specific configuration, after which an EEF pose offset is applied in the robot base frame. The left configuration applies a translation of $\Delta\mathbf{p} = [0, 0.10, 0]\,\mathrm{m}$ and a yaw rotation of $+45°$, whereas the right configuration applies $\Delta\mathbf{p} = [0, -0.10, 0]\,\mathrm{m}$ and a yaw rotation of $-45°$. All other evaluation conditions remain unchanged. We evaluate Stack Blocks, Erase Mark, and Stack Three Cups under both types of spatial shift, conducting five rollouts for each task and configuration.

As shown in Figure 6, OVOA-VLA improves the average success rate from 26.7% to 60.0% under camera viewpoint shifts and from 30.0% to 56.7% under robot initialization shifts. It outperforms $\pi_{0.5}$ across all six task–shift combinations, with the largest gains observed on Stack Three Cups under camera viewpoint shifts and Stack Blocks under robot initialization shifts. These results show that the improvement is not limited to a single type of spatial shift.

E.4 Additional Real-World Qualitative Results

We provide additional qualitative examples to complement the quantitative real-world evaluation. Figure 7 presents successful executions across all six tasks under the in-distribution setting, while Figure 8 shows successful executions from perturbed robot initializations.

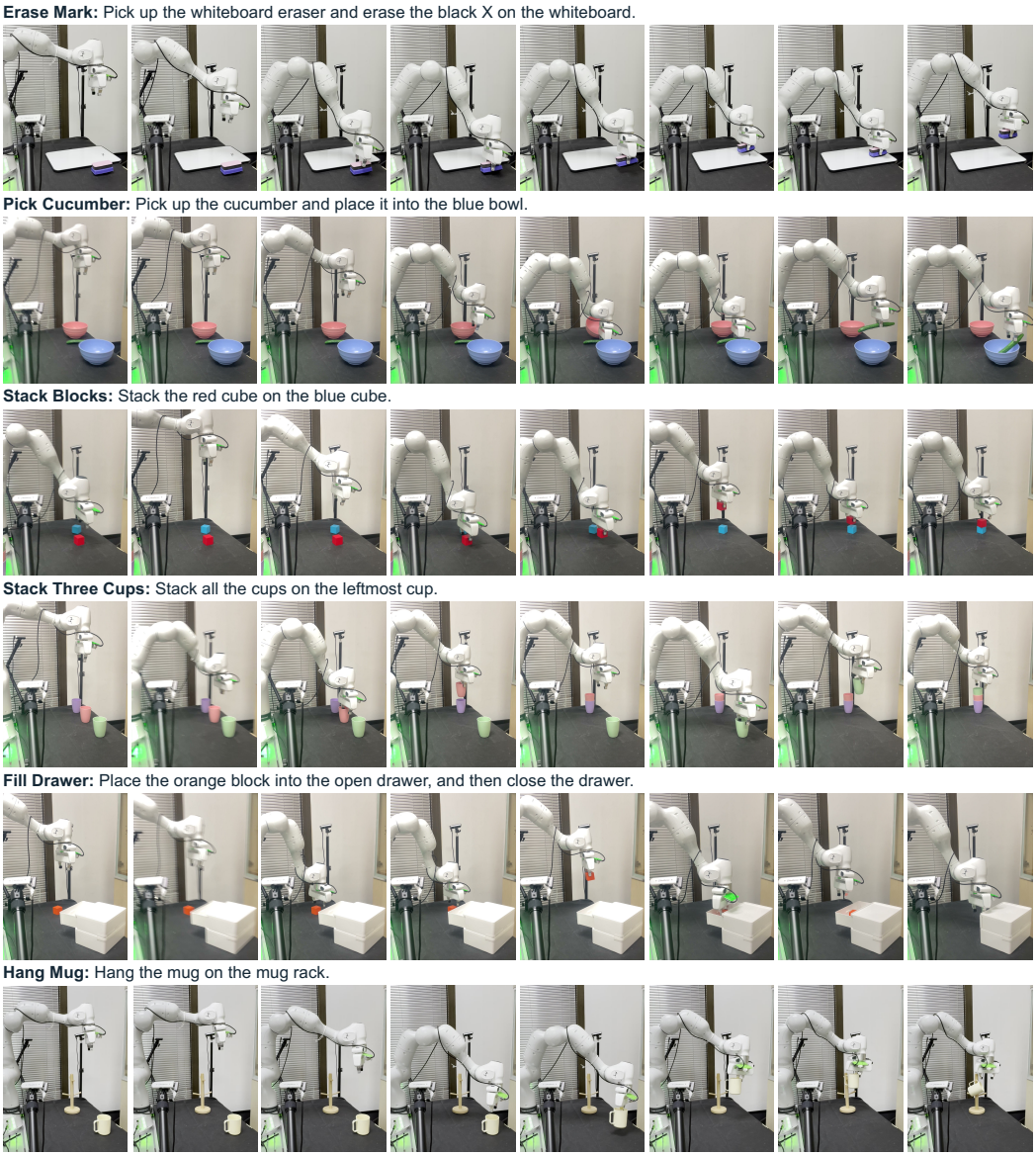


Figure 7: **Qualitative results under in-distribution evaluation.** Representative successful OVOA-VLA rollouts across the six evaluated real-world tasks.

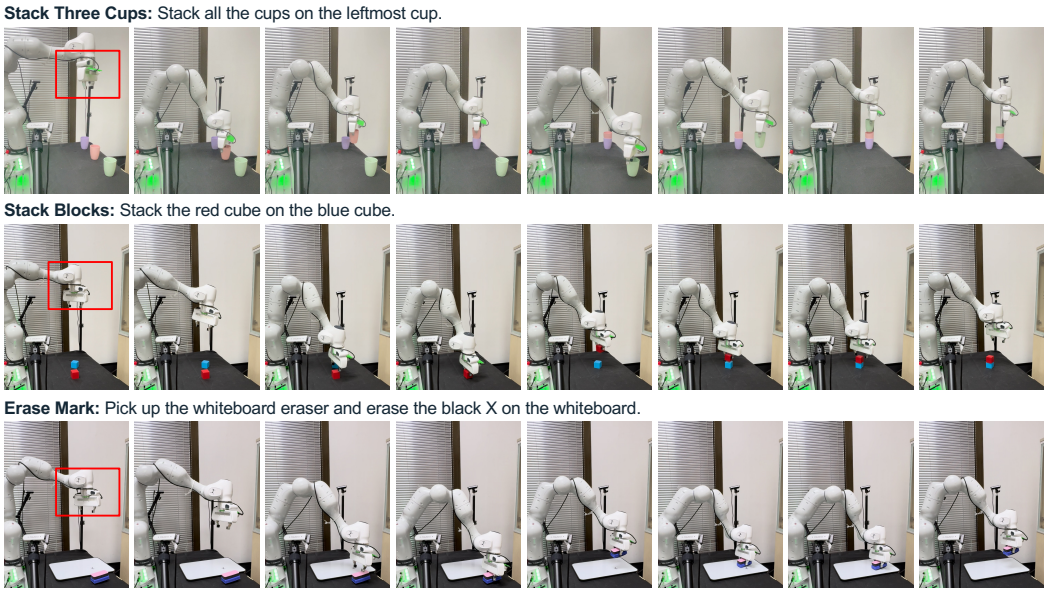


Figure 8: **Qualitative results under robot initialization shifts.** Representative successful OVOA-VLA rollouts on Stack Blocks, Erase Mark, and Stack Three Cups. The red box in the first frame of each rollout highlights the perturbed initial EEF pose.

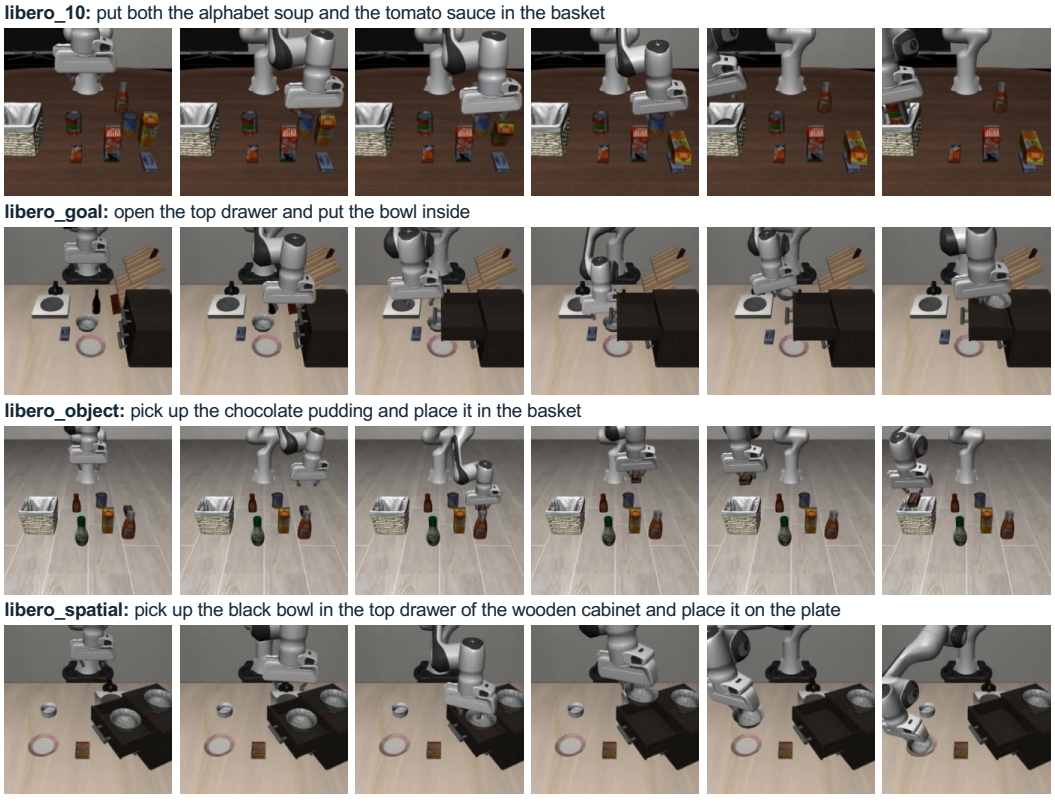


Figure 9: **Qualitative results on LIBERO.** Successful execution trajectories from the Spatial, Object, Goal, and Long task suites.



Figure 10: **Qualitative results on LIBERO-Plus.** Each row shows a successful rollout under one perturbation category: Camera, Robot, Language, Light, Background, Noise, or Layout. The first column presents the initial state from the corresponding unperturbed rollout for comparison, while the remaining six columns show keyframes from the successful perturbed rollout, including its initial frame.

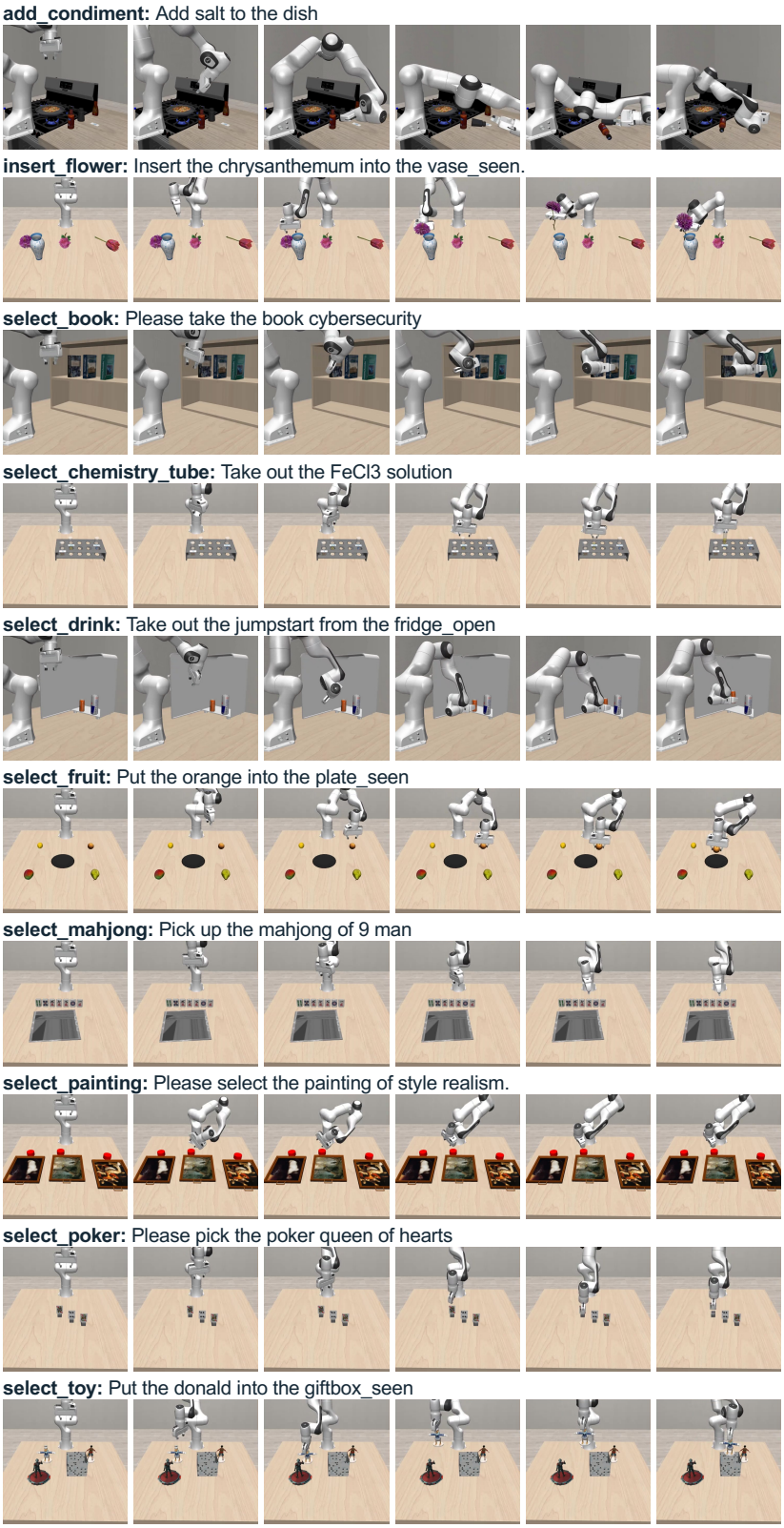


Figure 11: **Qualitative results on VLABench.** One successful execution trajectory for each of the ten evaluated tasks.

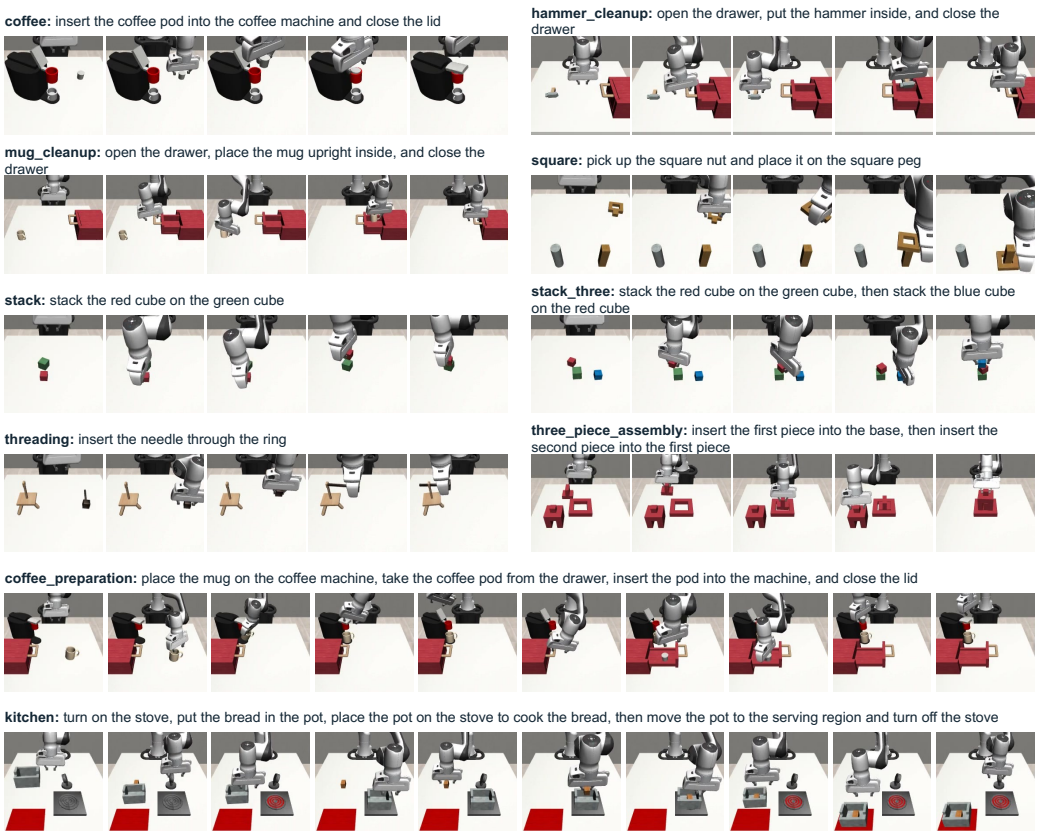


Figure 12: **Qualitative results on MimicGen.** One successful execution trajectory for each of the ten selected Panda D0 tasks.