REFLECTIVE VLA: IN-CONTEXT ACTION CONSEQUENCES MAKE VLAS GENERALIZE

Anonymous authors
Paper under double-blind review

ABSTRACT

Most vision-language-action (VLA) models are reactive: they predict actions from the current instruction and observation. However, deployment-specific factors such as camera geometry, robot calibration, and actuation bias can be difficult to identify from a single observation, limiting the policy's ability to adapt its behavior based on observed action consequences. We propose Reflective VLA, which conditions each decision on observation–action–consequence triplets collected during the policy's own task execution. Each triplet records what the robot observed and executed, and how the scene changed afterward, providing action-aligned feedback for subsequent decisions without test-time parameter updates. Architecturally, Reflective VLA routes all observation modalities through the VLM under shared attention, allowing the action expert to attend directly to past triplets and the current observation. A block-causal mask enables parallel multi-frame training without leakage and supports KV-cached inference. Reflective VLA maintains strong performance on LIBERO and improves success on SimplerEnv-Bridge. Under deployment perturbations on LIBERO-Plus and LIBERO-Plus-Hard, it improves average success by 5.4 and 4.2 percentage points over a matched reactive baseline, respectively. Hard evaluates deployment configurations outside the task-specific training distribution within camera and calibration perturbation families represented during training. Controlled ablations support the joint use of historical actions and observed consequences beyond visual history alone, and real-robot studies provide complementary evidence under camera shifts.

1 INTRODUCTION

Vision-language-action (VLA) systems build on pretrained vision-language backbones (Bai et al., 2025a;b; Beyer et al., 2024) and fine-tune on large-scale robot demonstrations (Open X-Embodiment Collaboration et al., 2024; AgiBot-World-Contributors et al., 2025) to produce language-conditioned control policies (Zitkovich et al., 2023; Kim et al., 2024; Black et al., 2025; Li et al., 2024a; NVIDIA et al., 2025). By unifying scene understanding, instruction following, and low-level control in one model, they have substantially broadened the range of manipulation tasks a generalist policy can perform. Yet despite training on large and diverse datasets, current VLAs still generalize poorly to deployment environments unseen during training (Fei et al., 2025; Zhang et al., 2025a).

We study deployment generalization: maintaining task success under changes in camera geometry, robot calibration, or actuation bias. Our OOD evaluation tests configurations outside the task-specific training distribution within perturbation families represented during training. For example, two deployments may present similar observations yet produce different motions in response to the same command. The *interaction context*—an observation, an executed action, and the resulting observation—captures this command–response relationship, providing evidence for adjusting subsequent actions to the current deployment. History-conditioned policies support state tracking and deployment adaptation (Liu et al., 2025; Shi et al., 2026; Kumar et al., 2021). Our empirical question is whether explicitly pairing executed action chunks with their aligned consequences improves control under these shifts, using history collected during task execution.

We therefore cast deployment adaptation in VLAs as an in-context learning (ICL) problem over observation–action–consequence triplets. Given a small prompt of such triplets, the policy uses interaction feedback to condition subsequent actions on observed responses, drawing on the context-

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.

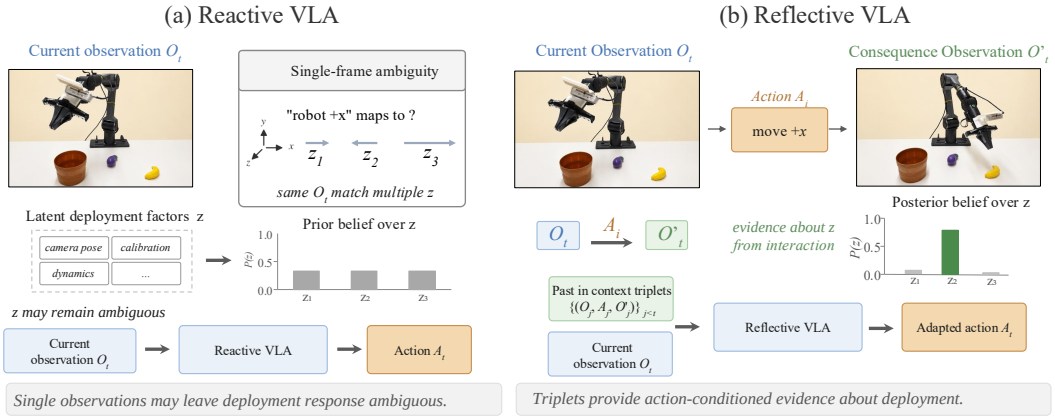


Figure 1: From reactive to reflective control. (a) Reactive VLA. A single observation O_t may be compatible with different deployment factors z, such as camera geometry or actuation bias, leaving the action-to-effect mapping ambiguous. (b) Reflective VLA. Past triplets $\{(O_j, A_j, O'_j)\}_{j<t}$ link executed actions to their observed consequences, providing evidence for adapting subsequent actions to the current deployment. The illustrated posterior $P(z \mid \text{context})$ provides a conceptual view of how interaction evidence can distinguish otherwise ambiguous deployments.

conditioned prediction paradigm of large language models (Brown et al., 2020; Xie et al., 2022). The context is accumulated during the policy's own task execution and supports adaptation without test-time parameter updates. Instantiating this in a VLM with a flow-matching action expert (Black et al., 2025; Physical Intelligence, 2025) raises two challenges. First, heterogeneous modalities—images, proprioception, and continuous action chunks—must be organized into a causal context that the action expert can access directly. Second, longer contexts increase training and inference costs, particularly when target frames are processed separately or historical representations are recomputed at every step.

We address these challenges with Reflective VLA. Multimodal observations and historical action chunks share the VLM token space, allowing the action expert to attend directly to past triplets and the current observation. A block-causal mask supervises all K sampled targets in a single forward pass while preventing each target from accessing its own action or consequence. The same causal structure supports KV-cached inference, reusing historical representations as new interactions enter the context.

In summary, our main contributions are:

1. We formulate deployment adaptation in VLAs as in-context learning from the policy's own observation–action–consequence history, using action-aligned feedback to adapt to sensing and control shifts without test-time parameter updates.
2. We introduce Reflective VLA, which equips a VLM with a flow-matching action expert to use shared multimodal interaction context, with block-causal multi-target training and KV-cached inference.
3. We evaluate nominal task performance and robustness on LIBERO, SimplerEnv-Bridge, and LIBERO-Plus, and demonstrate gains over a matched reactive baseline on OOD deployment configurations in LIBERO-Plus-Hard. Controlled ablations support the contribution of historical actions alongside matched visual feedback.

2 Related Work

Vision-language-action models. Generalist robot policies learn language-conditioned control from large-scale demonstrations (Reed et al., 2022; Brohan et al., 2023; Zitkovich et al., 2023; Kim et al., 2024; Octo Model Team et al., 2024). Action interfaces include autoregressive token prediction (Zitkovich et al., 2023; Kim et al., 2024; 2025; Pertsch et al., 2025) and continuous diffusion or flow-matching experts attached to a VLM (Black et al., 2025; Li et al., 2024a; Wu et al., 2026; Zheng

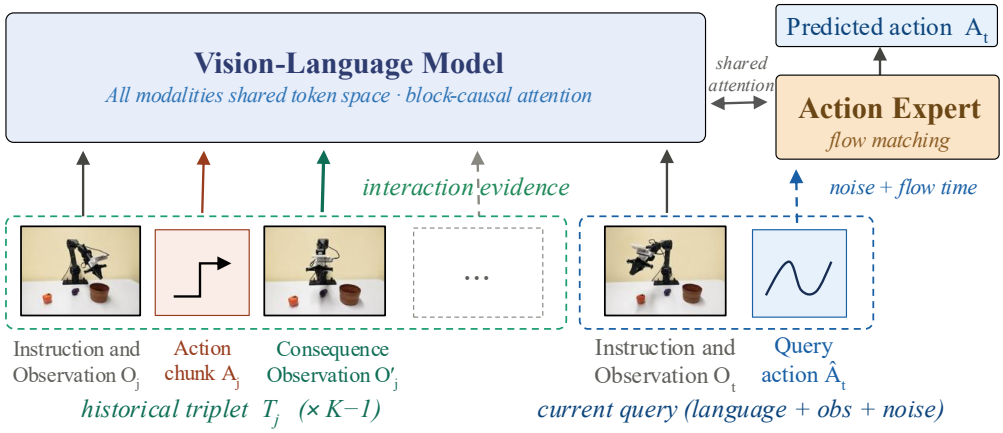


Figure 2: Reflective VLA: observation–action–consequence in-context learning. Past triplets $\{(O_j, A_j, O'_j)\}_{j=1}^{K-1}$ and the current observation O_t share a single VLM token sequence; a flow-matching action expert attends to this prefix and denoises $\hat{A}_t$ into the predicted chunk A_t. The aligned consequence O'_j carries the *interaction evidence* that exposes environment factors.

et al., 2026; NVIDIA et al., 2025). Reflective VLA adopts a continuous action expert and augments its multimodal prefix with interaction history.

Temporal context and memory for robot control. ACT, Diffusion Policy, and RDT use temporal observations or chunked actions for visuomotor control (Zhao et al., 2023; Chi et al., 2023; Liu et al., 2025). RoboFlamingo, 4D-VLA, and MemoryVLA incorporate visual history, spatiotemporal representations, or explicit memory (Li et al., 2024b; Zhang et al., 2025b; Shi et al., 2026). These approaches support state estimation and task tracking. Our ablations examine the effect of explicitly pairing historical action chunks with their aligned consequence observations.

In-context adaptation for sequential decision making. Context-conditioned adaptation connects to meta-RL (Duan et al., 2016; Finn et al., 2017; Rakelly et al., 2019) and sequence-based policies such as Decision Transformer, Prompt-DT, and algorithm distillation (Chen et al., 2021; Xu et al., 2022; Laskin et al., 2023). MTT studies how trajectory distributions enable generalization to new tasks (Raparthy et al., 2024). ICRT prompts robot policies with sensorimotor demonstrations (Fu et al., 2025); RICL adds demonstration-conditioned adaptation to pretrained VLAs (Sridhar et al., 2025b); and REGENT combines retrieval with in-context policy learning (Sridhar et al., 2025a). These works motivate distinguishing the source and purpose of context. Our context consists of the policy's own task-execution history and provides evidence about deployment conditions without test-time parameter updates.

Adaptation to deployment conditions. RMA uses a separate adaptation module to infer deployment conditions from history (Kumar et al., 2021). OC-VLA addresses viewpoint variation by expressing actions in camera coordinates using calibrated extrinsics (Zhang et al., 2026). Concurrent work, ICWM (Wang et al., 2026), also explores adaptation through self-generated interaction histories. ICWM collects task-agnostic interactions before task execution, whereas Reflective VLA continually incorporates observation–action–consequence triplets collected during task execution, using a flow-matching action expert and block-causal multi-frame training.

3 Method

3.1 Preliminary: Standard Reactive VLA Formulation

At control step t, let $\mathcal{L}$ denote the language instruction, $\mathcal{O}_t$ the current multimodal observation, and $A_t = [a_t, \ldots, a_{t+C-1}]$ an action chunk of horizon C. The observation may include third-person images, wrist images, and proprioceptive states. A standard reactive VLA predicts the next action

chunk from only the current instruction and observation:

$$A_t \sim \pi_\theta(\cdot \mid \mathcal{L}, \mathcal{O}_t). \tag{1}$$

3.2 From Reactive Control to In-Context Adaptation

Let z summarize deployment-specific camera geometry, calibration, and actuation bias. A single observation need not identify z, whereas interaction history $\mathcal{H}$ supplies evidence through observed action effects. This motivates the posterior-predictive view

$$\pi(A_t \mid \mathcal{L}, \mathcal{O}_t, \mathcal{H}) = \int \pi(A_t \mid \mathcal{L}, \mathcal{O}_t, z)\, P(z \mid \mathcal{H})\, dz. \tag{2}$$

This view provides conceptual motivation for conditioning on feedback (Xie et al., 2022). Reflective VLA predicts actions directly; it does not explicitly estimate z, fit a dynamics model, or evaluate this integral.

The key requirement is that $\mathcal{H}$ is diagnostic of z. Under an idealized Markov view, if $\mathcal{H}$ consists of interaction triplets, the posterior factorizes as

$$P(z \mid \mathcal{H}) \propto P(z) \prod_i P_{\text{env}}(\mathcal{O}'_i \mid \mathcal{O}_i, A_i, z), \tag{3}$$

where $\mathcal{O}'_i$ is the observation after executing action chunk A_i. This interpretation requires observable consequences, executed actions that distinguish relevant deployment factors through their effects, and history consistent with the current deployment. Factors inducing identical consequence distributions over the visited contexts cannot be distinguished by these triplets alone. Observations may themselves reveal deployment cues; $\mathcal{O}'_i$ adds action-conditioned evidence. This posterior view motivates the conditioning design: empirical gains do not establish general identifiability of z or exact Bayesian inference.

Reflective VLA therefore conditions each decision on interaction context with explicit consequences:

$$A_t \sim \pi_\theta\big(A_t \mid \mathcal{L}, \mathcal{H}, \mathcal{O}_t\big), \quad \mathcal{H} = \big\{(\mathcal{O}_i, A_i, \mathcal{O}'_i)\big\}_{i=1}^{K-1}, \tag{4}$$

where $\mathcal{H}$ is the observation–action–consequence context.

3.3 Reflective VLA

Observation–action–consequence context. Reflective VLA represents recent interaction history as structured observation–action–consequence triplets. Each triplet stores the observation before an action chunk, the action chunk executed from that observation, and the resulting observation after the chunk completes. Since the policy predicts C-step action chunks, we define the consequence observation as $\mathcal{O}'_i = \mathcal{O}_{i+C}$ rather than the immediate next frame $\mathcal{O}_{i+1}$. This action-aligned consequence better captures the visible effect of the executed action, such as end-effector displacement and residual control error.

At training time, A_i is the demonstration action chunk and $\mathcal{O}'_i$ the observation after the chunk horizon; at deployment, A_i is the chunk actually executed and $\mathcal{O}'_i$ the subsequent observation collected from the environment. Thus, each stored triplet represents a realized interaction rather than only a planned transition. We embed each previous action chunk into eight tokens in the VLM token space via a learned projection g_A, and write the resulting prefix tokens simply as A_i. The i-th context element is then $T_i = (\mathcal{L}, \tau_i, \mathcal{O}_i, A_i, \mathcal{O}'_i)$, where the language instruction $\mathcal{L}$ is re-inserted at the start of every triplet so that each unit is a self-contained $(\mathcal{L}, \mathcal{O}, A, \mathcal{O}')$ block, matching the per-triplet structure shown in Figure 3. We refer to these as causal triplets because the action token links a pre-action observation to its observed post-action consequence.

Multimodal prompt construction. Given historical milestones $m_1, \ldots, m_{K-1}$ and the current query timestep t, Reflective VLA packs past triplets and the current observation—each prefixed by the language instruction $\mathcal{L}$—into a single multimodal sequence:

$$X_{\text{in}} = \Big[T_{m_1}, \ldots, T_{m_{K-1}},\ (\mathcal{L}, \tau_t, \mathcal{O}_t)\Big]. \tag{5}$$

The historical triplets provide evidence about how the current deployment responds to actions, while the final $(\mathcal{L}, \tau_t, \mathcal{O}_t)$ block serves as the query for predicting the next action chunk. The current target action is not inserted into the prefix; it is generated by the continuous action expert.

Shared-attention architecture. For the action decoder to use interaction context, all observation modalities must be visible to it. Some VLA architectures route only part of the observation through the VLM backbone or condition the action module on a compressed prefix (NVIDIA et al., 2025; Zheng et al., 2026), which can bottleneck reasoning over (O, A, O') structure. Reflective VLA therefore routes all observation modalities—third-person images, wrist images, and proprioception—into a shared VLM token sequence. Images are encoded by the visual backbone, while proprioceptive states and previous action chunks are projected into the token space with a learned non-linear head. A continuous flow-matching action expert is attached as a suffix that shares attention with the VLM prefix at every layer, following recent MoT-style VLA designs (Liang et al., 2025; Black et al., 2025; Wu et al., 2026). The action expert can thus attend directly to prior observations, prior actions, observed consequences, temporal indices, and the current observation when generating A_t.

Block-causal training. In-context conditioning lengthens each training sequence by a factor of K, so supervising each context position with an independent forward pass would scale training cost as $\mathcal{O}(K)$. We instead borrow the packed-sequence idea from LM training and supervise all K sampled frames jointly under a *block-causal* attention mask, which lets every sampled frame act as both context for later targets and as a prediction target itself within a single forward pass. Because the same mask supervises positions with $0, 1, \ldots, K-1$ preceding triplets, training naturally covers the full range of context lengths the model will encounter at deployment.

Sampling and packing. We sample K ordered frames $t_1 < \cdots < t_K$ from a trajectory and pack them into a single sequence: the first $K-1$ form the historical triplets and the K-th is the current query, but the block-causal mask supervises all K frames jointly. As reflected in T_i, $\mathcal{L}$ is re-inserted at the start of every triplet (and the query block) so each unit is a self-contained $(\mathcal{L}, \mathcal{O}, A, \mathcal{O}')$ structure, and special tokens demarcate triplet boundaries so the action expert can unambiguously parse where each unit begins and ends. Because adjacent action chunks are temporally smooth, a fixed inter-frame stride invites the policy to extrapolate A_{t_k} from past actions rather than learn from observed consequences; we therefore randomize the stride within a bounded range during training to break this shortcut.

Mask and objective. Each target frame t_k is realized by a query slot $\hat{A}_{t_k}$ in the suffix flow-matching expert; Figure 3 visualizes the resulting attention pattern as a row in the mask. The query attends only to

$$\mathcal{V}_{t_k} = \{T_{t_j} : j < k\} \cup \{(\mathcal{L}, \tau_{t_k}, \mathcal{O}_{t_k})\}, \tag{6}$$

i.e., all completed prior triplets (each carrying its own copy of $\mathcal{L}$) together with the current language-prefixed observation. It is masked from the prefix action token A_{t_k} and consequence $\mathcal{O}'_{t_k}$ within its own triplet—both observed only after A_{t_k} is executed—and from every subsequent triplet $\{T_{t_j} : j > k\}$; sibling query slots $\{\hat{A}_{t_j}\}_{j \neq k}$ are also mutually masked, so each prediction depends solely on prefix evidence. The training objective sums the flow-matching action loss over all valid targets:

$$\mathcal{L}_{\text{train}} = \sum_{k=1}^{K} \mathcal{L}_{\text{act}}\big(A_{t_k}; \mathcal{V}_{t_k}\big), \tag{7}$$

where $\mathcal{L}_{\text{act}}$ is the same flow-matching objective used by the base VLA. This yields dense multi-frame supervision in one forward pass while ensuring that each prediction is conditioned only on past completed interactions and the current observation. The first target t_1 has no preceding triplets and provides reactive supervision, while $t_2, \ldots, t_K$ provide ICL supervision with progressively longer context, as the staircase pattern along the diagonal of Figure 3 shows.

At deployment, Reflective VLA keeps a rolling buffer of the most recent $K-1$ triplets. Unlike training, where ground-truth action chunks populate the historical context, at inference each triplet stores the policy's own predicted chunk $\hat{A}_t$ together with the observation $\mathcal{O}_{t+C}$ actually reached after executing it—so the in-context prefix reflects the realized rollout rather than a teacher trajectory. Only newly arriving inputs are encoded; VLM-side keys and values for past triplets are reused. For fixed K, per-step cost remains bounded as the rollout grows, but increases with K (Section C.4).

Table 1: In-distribution evaluation on LIBERO and SimplerEnv-Bridge. Success rates (%); Avg denotes the unweighted mean over the displayed suites or tasks. LIBERO averages are recomputed from the four displayed suite rates, excluding LIBERO-90. OpenVLA and CogACT LIBERO results are reported by Shi et al. (2026). † denotes our reproduced reactive baseline $\pi_{0.5}$. "–" indicates the result is not provided.

	LIBERO					SimplerEnv-Bridge				
Method	Spatial	Object	Goal	Long	Avg	Spoon	Carrot	Cube	Eggplant	Avg
OpenVLA (Kim et al., 2024)	84.7	88.4	79.2	53.7	76.5	4.2	0.0	8.3	45.8	14.6
CoT-VLA (Zhao et al., 2025)	87.5	91.6	87.6	69.0	83.9	–	–	–	–	–
4D-VLA (Zhang et al., 2025b)	93.8	92.8	95.6	86.5	92.2	–	–	–	–	–
ThinkAct (Huang et al., 2025)	–	–	–	–	–	37.5	8.7	58.3	70.8	43.8
CogACT (Li et al., 2024a)	97.2	98.0	90.2	88.8	93.6	71.7	50.8	15.0	67.5	51.3
InternVLA-M1 (Chen et al., 2025)	98.0	**99.0**	93.8	92.6	95.9	87.5	67.9	31.3	**100.0**	71.7
π_0 (Black et al., 2025)	96.8	98.8	95.8	85.2	94.2	–	–	–	–	–
MemoryVLA (Shi et al., 2026)	98.4	98.4	96.4	93.4	96.7	75.0	75.0	37.5	**100.0**	71.9
GR00T N1.5 (NVIDIA et al., 2025)	–	–	–	–	–	82.0	72.0	54.0	63.0	67.8
$\pi_{0.5}$ (Physical Intelligence, 2025)	**98.8**	98.2	98.0	92.4	96.9	–	–	–	–	–
Reactive baseline $\pi^{\dagger}_{0.5}$	97.5	98.2	97.8	94.0	96.9	91.7	79.2	70.8	50.0	72.9
Reflective VLA (ours)	98.4	**99.0**	**98.2**	**94.6**	**97.6**	**95.8**	**83.3**	**79.2**	54.2	**78.1**

Table 2: Robustness under perturbations on LIBERO-Plus and LIBERO-Plus-Hard. Success rates (%) on LIBERO-Plus (7 standard perturbation categories) and LIBERO-Plus-Hard (2 additional harder shifts: Multi-camera shift (Camera†) and Robot calibration shift (Rob. Calib†)). Hard tests larger camera-pose shifts and command biases within perturbation families included in training (ranges in Table 4).

	LIBERO-Plus								LIBERO-Plus-Hard		
Method	Camera	Robot	Lang	Light	Bg	Noise	Layout	Avg	Camera†	Rob. Calib†	Avg
UniVLA (Bu et al., 2025)	1.8	46.2	69.6	69.0	81.0	21.2	31.9	45.8	–	–	–
π_0 (Black et al., 2025)	13.8	6.0	58.8	85.0	81.4	79.0	68.9	56.1	–	–	–
OpenVLA-OFT (Fei et al., 2025)	92.8	30.3	85.8	94.9	93.9	89.3	**77.6**	80.7	72.2	43.1	57.7
MemoryVLA (Shi et al., 2026)	93.1	42.1	84.2	92.1	92.7	89.1	77.5	81.5	72.1	49.2	60.7
Reactive baseline $\pi^{\dagger}_{0.5}$	90.0	50.0	92.9	92.0	85.8	90.2	75.0	82.3	74.0	55.2	64.6
Reflective VLA (ours)	**95.7**	**72.9**	92.2	92.1	92.7	**95.4**	73.0	**87.7**	**76.3**	**61.3**	**68.8**

4 Experiments

We evaluate Reflective VLA along three axes. First, we test whether adding interaction context preserves strong in-distribution performance on standard manipulation benchmarks. Second, we evaluate robustness under perturbations that change visual appearance, sensing, or embodiment-specific properties. Third, we ablate context composition and length to evaluate the contributions of historical actions, observed consequences, and the amount of interaction history.

4.1 Experimental Setup

Simulation benchmarks. We evaluate on four simulation settings: *(i)* LIBERO (Liu et al., 2023), the standard suite of 40 language-conditioned manipulation tasks across *Spatial*, *Object*, *Goal*, and *Long*, with a fixed camera and embodiment, used as our nominal in-distribution testbed; *(ii)* SimplerEnv-Bridge (Li et al., 2024c), which evaluates VLAs trained on the BridgeDataV2 (Walke et al., 2023) in the ManiSkill2 (Gu et al., 2023) simulator, providing a real-to-sim transfer setting; *(iii)* LIBERO-Plus (Fei et al., 2025), which extends LIBERO with seven perturbation categories spanning sensing, language, and robot–environment configuration; and *(iv)* LIBERO-Plus-Hard, our diagnostic extension targeting shifts in the action-to-observation mapping.

A context-diagnostic benchmark. LIBERO-Plus-Hard focuses on two perturbations to the action-to-observation mapping for which a single frame may provide limited deployment-specific evidence, motivating the use of interaction feedback:

Table 3: Ablations on the interaction context. Success rates (%) on Camera, Camera†, and Rob. Calib†; Avg denotes the unweighted mean. (a) Separately trained context variants at fixed K, with a reactive reference. The listed components denote explicit entries in the historical context; all variants additionally receive the current observation and instruction. (b) One fixed model evaluated at different context lengths K using (O, A, O') history; $K{=}1$ provides no history.

(a) Context composition (fixed K).

Context	Camera	Camera†	Rob. Calib†	Avg
Reactive (no history)	90.0	74.0	55.2	73.1
O	88.8	73.2	55.0	72.3
O, A	89.1	74.9	57.4	73.8
O, O'	88.5	73.3	55.1	72.3
O, A, O'	**95.7**	**76.3**	**61.3**	**77.8**

(b) Inference context length (same model).

K	Camera	Camera†	Rob. Calib†	Avg
1	90.0	74.0	55.2	73.1
2	93.8	74.6	57.4	75.3
4	**96.1**	75.7	58.4	76.7
8	95.7	**76.3**	**61.3**	**77.8**

- **Multi-camera shift** (Camera† in Table 2). We vary third-person camera azimuth, elevation, and distance together with wrist-image transformations. Past action chunks and their observed effects provide evidence about the changed action-to-pixel mapping.
- **Robot calibration shift** (Rob. Calib† in Table 2). We inject an episode-level translational bias into end-effector position commands, simulating a systematic command–response mismatch. Completed triplets expose discrepancies between commands and observed effects that can inform subsequent actions.

Generalization to OOD deployment configurations. Camera† and Rob. Calib† test extrapolation toward larger camera-pose shifts and command biases within perturbation families used in training (Table 4). Reflective VLA and the reactive baseline share the same augmented training data, allowing us to evaluate the benefit of execution history under these shifts (Section C.1).

Baselines. Our primary baseline, denoted $\pi^\dagger_{0.5}$, is a reproduced *reactive* $\pi_{0.5}$ that uses the same backbone, training data, and network parameters as Reflective VLA but with context length $K{=}1$, conditioning each prediction only on the current instruction and observation. Where available and protocol-compatible, we additionally report published results for OpenVLA (Kim et al., 2024), OpenVLA-OFT (Kim et al., 2025), UniVLA (Bu et al., 2025), CoT-VLA (Zhao et al., 2025), 4D-VLA (Zhang et al., 2025b), ThinkAct (Huang et al., 2025), CogACT (Li et al., 2024a), InternVLA-M1 (Chen et al., 2025), π_0 (Black et al., 2025), $\pi_{0.5}$ (Physical Intelligence, 2025), MemoryVLA (Shi et al., 2026), and GR00T N1.5 (NVIDIA et al., 2025).

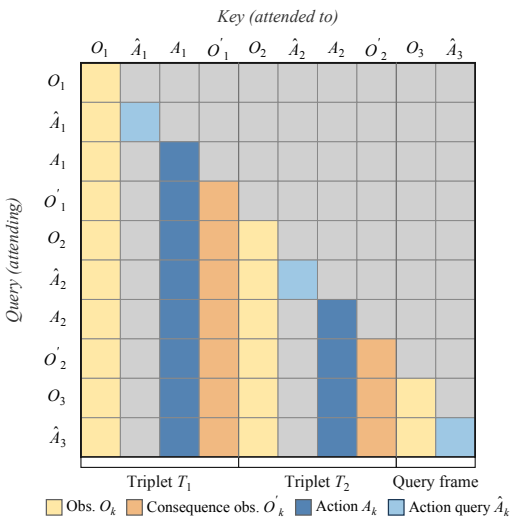


Figure 3: Block-causal mask. Each query $\hat{A}_k$ attends to $\mathcal{L}$, prior triplets $T_{<k}$, and $\mathcal{O}_k$, while its own A_k, $\mathcal{O}'_k$, future triplets and queries are masked, supervising all targets in one forward.

Implementation details. Following $\pi_{0.5}$ (Physical Intelligence, 2025), Reflective VLA adopts a Mixture-of-Transformers (Liang et al., 2025) design, pairing a pretrained VLM prefix with a flow-matching continuous action expert as the suffix under shared attention. Unless otherwise stated, we use action chunk size $C{=}10$, context length $K{=}8$, corresponding to seven historical triplets and one query observation, and bounded randomized milestone sampling. Additional architectural details, hyperparameters, and training settings are provided in Sections A and C.1.

4.2 Main Results

In-distribution performance. Table 1 shows that Reflective VLA preserves strong performance on LIBERO and SimplerEnv-Bridge. On LIBERO it reaches 97.6% average success over the four displayed suites, compared with 96.9% for the published $\pi_{0.5}$ and 96.7% for MemoryVLA, and improves over the matched reactive baseline by 0.7 percentage points. On SimplerEnv-Bridge, it reaches 78.1% average success versus 72.9% for the reactive baseline, a gain of 5.2 points. Performance on *Long* also increases slightly (94.0% $\rightarrow$ 94.6%). These results support the use of interaction context while preserving nominal task performance.

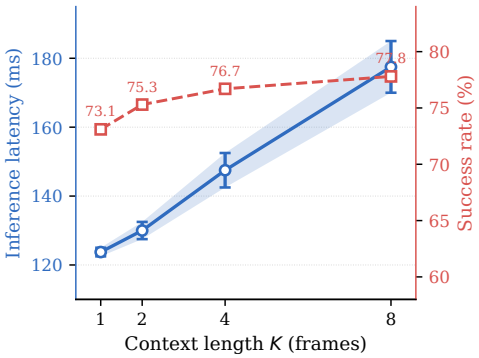


Figure 4: Latency–accuracy trade-off across context length K on the perturbation subset.

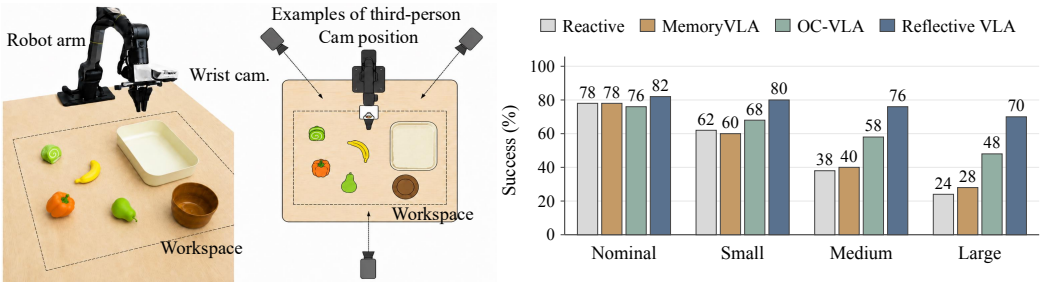
Robustness under perturbations. Table 2 evaluates seven standard LIBERO-Plus perturbation categories and two harder shifts; MemoryVLA is reproduced on the same training data as ours. On LIBERO-Plus, Reflective VLA achieves 87.7% average success versus 82.3% for the matched reactive baseline, 80.7% for OpenVLA-OFT, and 81.5% for MemoryVLA. In this evaluation, it improves on five categories, with the largest gains under *Robot* (+22.9 pp), *Background* (+6.9 pp), and *Camera* (+5.7 pp); *Language* and *Layout* decrease by 0.7 and 2.0 points. The separate five-training-seed experiment in Table 6 also shows a positive average gain (89.0% vs. 82.6%; +6.4 pp), with a Robot gain of 22.9 points. Its standard deviations describe that repeated experiment and are not error bars for the point estimates in Table 2.

Generalization to OOD deployment configurations. LIBERO-Plus-Hard evaluates camera and calibration configurations outside the task-specific training distribution. On this subset, Reflective VLA reaches 68.8% average success, outperforming MemoryVLA (60.7%; +8.1 pp) and the matched reactive baseline (64.6%; +4.2 pp). Under *Multi-camera shift* (Camera†), it improves from 74.0% to 76.3%; under *Robot calibration shift* (Rob. Calib†), it improves from 55.2% to 61.3%. Both conditions change the relationship between actions and observed effects. With training data shared by the two policies, these gains show that interaction-conditioned control improves task success under the evaluated distribution shifts. The matched-visual-input ablation below tests whether historical actions contribute to this benefit.

Context composition. Table 3a examines historical actions and consequence observations as complementary context components. Variants are trained separately at fixed K; all receive the current observation and instruction, and history-conditioned variants use the same sampling scheme. The (O, O') variant includes pre-action and consequence observations but omits the executed action. It therefore records an observed change without explicitly linking that change to the executed command. Providing the corresponding action record in the full (O, A, O') triplet improves average success from 72.3% to 77.8%, supporting the value of pairing observed consequences with executed actions. Conversely, (O, A) omits explicit per-triplet consequence entries and reaches 73.8%, 4.0 percentage points below the full model. Together, these comparisons support the joint use of actions and their observed consequences as interaction feedback in the evaluated context variants.

Context length. Table 3b varies inference-time context length for one fixed model. At $K{=}1$, it receives only the current observation and instruction; larger K supplies up to $K{-}1$ completed triplets. Average success rises from 73.1% at $K{=}1$ to 75.3%, 76.7%, and 77.8% at $K{=}2, 4, 8$. Most of the gain is obtained by $K{=}4$, with a further 1.1-point improvement at $K{=}8$, showing that the same trained model benefits from additional interaction history.

Latency–accuracy trade-off. Figure 4 reports per-step latency against success rate during inference. Since past triplets are encoded once and reused, latency grows sub-linearly in K: $K{=}8$ is only $1.43\times$ slower than $K{=}1$ (178 vs. 124 ms) despite an $8\times$ longer prompt, while $K{=}4$ already recovers most



(a) Real robot setup (b) Camera placement protocol (c) Average success across camera-pose shifts

Figure 5: Real-world setup and results. (a) Piper robot and cameras. (b) Example third-person camera placements; demonstrations use ten of 25 positions. (c) Success rates averaged over Box and Bowl under four camera-pose shift conditions. OC-VLA uses calibrated camera extrinsics; Reflective VLA uses interaction history without camera extrinsics.

of the accuracy gain at $1.19\times$ baseline latency—a practical operating point when tighter control rates are required. We cap K at 8 because longer contexts exceed our GPU memory budget during block-causal training; KV-cached inference itself can scale further.

4.3 Real-World Experiments

We evaluate real-world cross-camera generalization on two tabletop tasks: placing an object into a large square box and into a smaller bowl, which requires tighter spatial alignment. We collect 500 demonstrations at ten of 25 third-person camera placements. Nominal uses these ten training placements; Small, Medium, and Large each use five held-out placements with offsets of 0.15, 0.30, and 0.60 from the nearest training placement, normalized by the standard deviation of the training camera distribution. Each placement is evaluated with five Box and five Bowl rollouts, giving 250 rollouts per method in total. We compare the reactive baseline, MemoryVLA (Shi et al., 2026), the calibration-aware OC-VLA (Zhang et al., 2026), and Reflective VLA under Nominal, Small, Medium, and Large camera-pose shifts (Figure 5c). Setup details and the complete numerical comparison are in Section D and Table 9.

Reflective VLA achieves average success rates of 82%, 80%, 76%, and 70% across the four conditions. Under Large shifts, it reaches 70% compared with 24% for the reactive baseline, 28% for MemoryVLA, and 48% for OC-VLA. OC-VLA improves robustness using calibrated camera extrinsics, while Reflective VLA exceeds it by 12, 18, and 22 percentage points under Small, Medium, and Large shifts, respectively, without receiving camera extrinsics. These results support the value of interaction history for adaptation to the evaluated camera changes. Additional real-robot qualitative sequences are shown in Figure 7.

5 Conclusion

Reflective VLA uses its own observation–action–consequence history to adapt without test-time parameter updates. Experiments show improved task success under the evaluated sensing and control shifts, and a matched-visual-input ablation supports the contribution of historical actions. These results provide behavioral evidence for feedback-conditioned adaptation. Its usefulness depends on informative actions, observable consequences, and history that remains relevant to the current deployment.

Limitations and Future Work. Reflective VLA starts reactively, assumes consequences are observable within each action chunk, and is limited to $K{=}8$ by training memory. Misaligned actions and consequences, stale history after deployment changes, or degraded consequence observations can make context misleading and performance worse than a reactive policy. Degradation under history corruption therefore reveals a failure mode, not by itself a unique adaptation mechanism. Real-world validation covers two tabletop tasks with few trials. Future work includes detecting unreliable history, mitigating history shortcuts, and broader validation across contexts, horizons, data, and embodiments.

AI USE STATEMENT

OpenAI Codex assisted with discussing research ideas, evaluation protocols, and ablation designs; organizing author-provided experimental results; and drafting descriptions of the analyses. It also assisted with literature search, editing related-work descriptions, checking bibliographic metadata, manuscript revision, migration to the ICLR 2027 LaTeX template, formatting and cross-reference checks, preparation of figures from existing results and real-robot recordings, and drafting the submission statements. Reference metadata were cross-checked against original papers, official publication records, and authors' official resources during this AI-assisted preparation. The authors take responsibility for the final text, claims, experimental results, and artifacts, including content prepared with AI assistance.

ETHICS STATEMENT

This work studies vision-language-action policies for robot manipulation. Improved generalization does not establish safe behavior outside the evaluated conditions: deployment under distribution shift can still produce incorrect or unsafe physical actions. Physical deployment should therefore include human supervision, appropriate workspace restrictions, and independent safety checks. The scope and limitations of the real-world evaluation are described in Appendix D and Section 5.

REPRODUCIBILITY STATEMENT

Section 3 describes the conditioning interface and training objective. Appendix A provides implementation and context-construction details; Appendix C describes training settings and evaluation protocols; Appendix B specifies the diagnostic perturbations; and Appendix D documents the hardware, tasks, and camera-placement protocol. Additional resource information is discussed in Appendix E, and limitations are discussed in Section 5.

REFERENCES

AgiBot-World-Contributors, Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, et al. AgiBot World Colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bošnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier Henaff, Xi Xiong, Radu Soricut, Jeremiah Harmsen, and Xiaohua Zhai. PaliGemma: A versatile 3B VLM for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. In *RSS*, 2025. doi: 10.15607/RSS.2025.XXI.010.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. In *RSS*, 2023. doi: 10.15607/RSS.2023.XIX.025.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, volume 33, 2020.

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 *RSS*, 2025. doi: 10.15607/RSS.2025.XXI.014.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *NeurIPS*, 2021.

Xinyi Chen, Yilun Chen, Yanwei Fu, Ning Gao, Jiaya Jia, Weiyang Jin, Hao Li, Yao Mu, Jiangmiao Pang, Yu Qiao, Yang Tian, Bin Wang, Bolun Wang, Fangjing Wang, Hanqing Wang, Tai Wang, Ziqin Wang, Xueyuan Wei, Chao Wu, Shuai Yang, Jinhui Ye, Junqiu Yu, Jia Zeng, Jingjing Zhang, Jinyu Zhang, Shi Zhang, Feng Zheng, Bowen Zhou, and Yangkun Zhu. InternVLA-M1: A spatially guided vision-language-action framework for generalist robot policy. *arXiv preprint arXiv:2510.13778*, 2025.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *RSS*, 2023. doi: 10.15607/RSS.2023.XIX.026.

Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. LIBERO-Plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

Max Fu, Huang Huang, Gaurav Datta, Lawrence Yunliang Chen, Will Panitch, Fangchen Liu, Hui Li, and Ken Goldberg. ICRT: In-context imitation learning via next-token prediction. In *IEEE International Conference on Robotics and Automation*, pp. 5937–5944, 2025. doi: 10.1109/ICRA55743.2025.11128272. URL https://doi.org/10.1109/ICRA55743.2025.11128272.

Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, Xiaodi Yuan, Pengwei Xie, Zhiao Huang, Rui Chen, and Hao Su. ManiSkill2: A unified benchmark for generalizable manipulation skills. In *International Conference on Learning Representations*, 2023.

Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. ThinkAct: Vision-language-action reasoning via reinforced visual latent planning. In *NeurIPS*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Pannag 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 *CoRL*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *RSS*, 2025. doi: 10.15607/RSS.2025.XXI.017.

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. RMA: Rapid motor adaptation for legged robots. In *Proceedings of Robotics: Science and Systems*, 2021. doi: 10.15607/RSS.2021.XVII.011. URL https://www.roboticsproceedings.org/rss17/p011.html.

Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Hansen, Angelos Filos, Ethan Brooks, Maxime Gazeau, Himanshu Sahni, Satinder Singh, and Volodymyr Mnih. In-context reinforcement learning with algorithm distillation. In *ICLR*, 2023.

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. *arXiv preprint arXiv:2411.19650*, 2024a.

Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, Hang Li, and Tao Kong. Vision-language foundation models as effective robot imitators. In *ICLR*, 2024b.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024c.

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.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *NeurIPS*, 2023.

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. In *ICLR*, 2025.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. In *RSS*, 2024. doi: 10.15607/RSS.2024.XX.090.

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, et al. Open x-embodiment: Robotic learning datasets and rt-x models. In *ICRA*, 2024.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

Physical Intelligence. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *ICML*, 2019.

Sharath Chandra Raparthy, Eric Hambro, Robert Kirk, Mikael Henaff, and Roberta Raileanu. Generalization to new sequential decision making tasks with in-context learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42138–42158. PMLR, 2024. URL https://proceedings.mlr.press/v235/raparthy24a.html.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gómez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Giménez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *Transactions on Machine Learning Research*, 2022.

Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. Memoryvla: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. In *ICLR*, 2026.

Kaustubh Sridhar, Souradeep Dutta, Dinesh Jayaraman, and Insup Lee. REGENT: A retrieval-augmented generalist agent that can act in-context in new environments. In *International Conference on Learning Representations*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/6237474e5a1c58a193af179b2434a86e-Abstract-Conference.html.

Kaustubh Sridhar, Souradeep Dutta, Dinesh Jayaraman, and Insup Lee. RICL: Adding in-context adaptability to pre-trained vision-language-action models. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 5022–5038. PMLR, 2025b. URL https://proceedings.mlr.press/v305/sridhar25a.html.

Homer Rich Walke, Kevin Black, Tony Z. Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, Abraham Lee, Kuan Fang, Chelsea Finn, and Sergey Levine. BridgeData V2: A dataset for robot learning at scale. In *Conference on Robot Learning*, volume 229, pp. 1723–1736, 2023.

Siyin Wang, Junhao Shi, Senyu Fei, Zhaoyang Fu, Li Ji, Jingjing Gong, and Xipeng Qiu. In-context world modeling for robotic control. *arXiv preprint arXiv:2606.26025*, 2026. URL https://arxiv.org/abs/2606.26025.

Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, et al. A pragmatic vla foundation model. *arXiv preprint arXiv:2601.18692*, 2026.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *ICLR*, 2022.

Mengdi Xu, Yikang Shen, Shun Zhang, Yuchen Lu, Ding Zhao, Joshua B. Tenenbaum, and Chuang Gan. Prompting decision transformer for few-shot policy generalization. In *ICML*, 2022.

Hongyin Zhang, Shuo Zhang, Junxi Jin, Qixin Zeng, Runze Li, and Donglin Wang. Robustvla: Robustness-aware reinforcement post-training for vision-language-action models. *arXiv preprint arXiv:2511.01331*, 2025a.

Jiahui Zhang, Yurui Chen, Yueming Xu, Ze Huang, Yanpeng Zhou, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, Xingyue Quan, Hang Xu, and Li Zhang. 4d-vla: Spatiotemporal vision-language-action pretraining with cross-scene calibration. In *NeurIPS*, 2025b.

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, 2026. doi: 10.1609/aaai.v40i22.38947. URL https://ojs.aaai.org/index.php/AAAI/article/view/38947.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, Ankur Handa, Tsung-Yi Lin, Gordon Wetzstein, Ming-Yu Liu, and Donglai Xiang. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *CVPR*, pp. 1702–1713, 2025.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *RSS*, 2023. doi: 10.15607/RSS.2023.XIX.016.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *ICLR*, 2026.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *CoRL*, 2023.

APPENDIX

This appendix provides the implementation and experimental details needed to reproduce Reflective VLA. Section A describes the architecture components, triplet construction, context-buffer behavior, block-causal masking, and flow-matching inference procedure. Section B specifies the two diagnostic perturbation families used in LIBERO-Plus-Hard. Section C details the training data construction, optimization settings, and evaluation protocol for LIBERO, LIBERO-Plus, LIBERO-Plus-Hard, and SimplerEnv-Bridge. Section D documents the real-world robot setup, control interface, tasks, and camera-placement protocol. Section E summarizes reproducibility resources and external assets.

A IMPLEMENTATION DETAILS

A.1 ARCHITECTURE COMPONENTS

Reflective VLA uses the same model family as the reactive baseline, with an interaction-conditioned input interface. Training steps, per-GPU batch sizes, and context lengths are reported separately for each benchmark and method in Table 5. We implement it with a Mixture-of-Transformers (MoT) architecture: a pretrained VLM encodes the language and multimodal observation prefix, and a continuous flow-matching transformer action expert is attached as a suffix with shared self-attention to the prefix. We instantiate the VLM with either PaliGemma-3B (Beyer et al., 2024) or Qwen3-VL-2B (Bai et al., 2025a), and use an action expert with hidden dimension 1024. Third-person and wrist images are processed through the native VLM visual pathway, while proprioceptive states and historical action chunks are projected into the token space by lightweight two-layer nonlinear fully connected projectors. Each historical action chunk is represented by eight learned tokens. The matched reactive baseline uses the same backbone, projectors, action expert, optimizer, and data, but sets $K{=}1$ so that no historical triplets are provided.

A.2 TRIPLET CONSTRUCTION AND CONTEXT BUFFER

For each control step t, the query input contains the instruction, the current multimodal observation, and a bounded history of completed interaction triplets. The current observation comprises the third-person view, left/right wrist views (when available), and the proprioceptive state. For a chunk horizon C, each historical triplet is stored as

$$T_i = \big(\tau_i, \mathcal{O}_i, A_i, \mathcal{O}_{i+C}\big), \tag{8}$$

where $A_i = [a_i, \ldots, a_{i+C-1}]$ is the executed action chunk, projected by g_A into eight learned action tokens. We use $C{=}10$ for LIBERO and SimplerEnv-Bridge, and $C{=}30$ for the real-world experiments.

During training, triplets are sampled only from completed past interactions of the same episode, so each consequence $\mathcal{O}_{i+C}$ strictly precedes the query. To prevent the model from extrapolating the target action from a fixed temporal pattern of neighboring actions, we add a randomized stride of $[0, 15]$ environment steps to the backward spacing.

At inference, the policy maintains a rolling FIFO buffer of completed triplets from the current episode: after each executed chunk, the newly formed triplet is appended and the oldest is evicted once the buffer is full. Triplets are drawn from the buffer with a fixed backward stride for deterministic input layout, and at the start of an episode the model simply conditions on whatever context is available.

A.3 PROMPT PACKING DETAILS

VLM input template

```
<|im_start|>user {instruction}
# historical triplets
<|frame_start|> <Third-Person OBS_i> <Wrist OBS_i> <PROPRIO_i>
<|action_start|><ACTION_i><|action_end|>
<Third-Person OBS_{i+C}> <Wrist OBS_{i+C}> <PROPRIO_{i+C}> <|frame_end|>
<|frame_sep|>   ···   <|frame_sep|>
# current query, no target action or consequence
<|frame_start|> <Third-Person OBS_t> <Wrist OBS_t> <PROPRIO_t>
```

B LIBERO-PLUS-HARD SPECIFICATION

B.1 PERTURBATION OVERVIEW

LIBERO-Plus-Hard extends LIBERO-Plus with two diagnostic perturbation families targeting factors that are hard to identify from a single frame but directly govern the action-to-observation mapping: camera geometry and robot calibration. We denote them Camera† and Rob. Calib† in the result tables. Task semantics, objects, and language instructions are unchanged.

Hard evaluates extrapolation toward larger camera-pose shifts and command biases within perturbation families represented in task-specific training (Table 4). Reflective VLA and its reactive baseline use the same augmented training data (Section C.1).

For each rollout, the perturbation parameters are sampled once at the start of the episode and held fixed throughout. All methods share the same task initializations, seeds, and sampled perturbation instances, and the success criterion follows the original LIBERO task-completion signal. The sampled perturbation parameters are not provided to the policy.

Table 4: Training and test ranges for LIBERO-Plus-Hard. Angular offsets and camera-distance scales are relative to the nominal third-person camera pose; translational biases are specified per Cartesian axis. Wrist-image transformations are included in both splits.

Perturbation parameter	Training	Hard evaluation
Third-person azimuth	$[-15^\circ, 15^\circ]$	$[-60^\circ, -15^\circ] \cup [15^\circ, 60^\circ]$
Third-person elevation	$[-15^\circ, 15^\circ]$	$[-60^\circ, -15^\circ] \cup [15^\circ, 60^\circ]$
Camera-distance scale	$[1.0, 1.5]$	$[1.5, 2.0]$
Wrist-image transformation	Identity, horizontal/vertical flip, 180° rotation	
Translation bias (mm/axis)	$[-5, 5]$	$[-10, -5] \cup [5, 10]$

B.2 MULTI-CAMERA SHIFT

The multi-camera shift changes third-person azimuth, elevation, and distance according to Table 4, while leaving task, object layout, and robot dynamics unchanged. Training and evaluation use the same wrist-image transformation family: identity, horizontal flip, vertical flip, or 180° rotation. Thus, the configuration shift increases third-person angular and distance perturbations in the presence of wrist-image transformations. Past actions and their observed effects provide evidence about the resulting action-to-pixel mapping.

B.3 ROBOT CALIBRATION SHIFT

The robot calibration shift adds an episode-level translational bias to every absolute end-effector position command before stepping the environment; the gripper command is unchanged. Biases are sampled independently along each Cartesian axis, with magnitudes increasing from at most 5 mm during training to 5–10 mm at test time (Table 4).

Table 5: Training hyperparameters. Batch size is reported per GPU; all reported runs use 8 H20 GPUs.

Setting	Steps	Batch/GPU	K
LIBERO reactive baseline	90k	32	1
LIBERO Reflective VLA	50k	4	8
LIBERO-Plus / Hard reactive	90k	32	1
LIBERO-Plus / Hard Reflective VLA	90k	4	8
SimplerEnv-Bridge reactive	160k	32	1
SimplerEnv-Bridge Reflective VLA	160k	6	4

C Experimental Protocol

C.1 Training details

Training data construction. For the standard LIBERO and SimplerEnv-Bridge experiments, we use the official LIBERO demonstrations and BridgeData V2 trajectories, with action targets converted to absolute end-effector control and a chunk horizon of $C{=}10$.

For LIBERO-Plus, we use the released LIBERO-Plus training set and convert the trajectories to absolute end-effector control via replay. For LIBERO-Plus-Hard, we follow the same replay-based generation pipeline on the original LIBERO demonstrations, additionally injecting camera-pose shifts and commanded-action biases from the training ranges in Table 4 during replay. We retain successfully completed replay trajectories for this training set. We merge the two sets and train a single model on the union, yielding 43,546 trajectories in total.

Deployment splits and scope. Camera† and Rob. Calib† in Table 2 evaluate the larger-magnitude test ranges in Table 4. The configuration shift concerns third-person camera angles and distance for Camera†, and per-axis translation bias for Rob. Calib†; wrist-image transformations occur in both camera splits. Both Reflective VLA and the reactive baseline are trained on the same merged data, so the comparison measures the benefit of execution history under these shifts without test-time parameter updates. The training-distribution reference concerns task-specific training and augmentation, without making claims about coverage in backbone pretraining. The real-world camera-placement protocol is defined separately in Section D.

Optimization. All models are trained with AdamW ($\beta_1{=}0.9$, $\beta_2{=}0.95$, weight decay 0.01, gradient clipping at norm 1.0) under bf16 mixed precision with DeepSpeed ZeRO-2. The action expert and projectors use a peak learning rate of 10^{-4}; the VLM uses a $0.1\times$ multiplier and is held frozen for the first 2,000 training steps, after which it follows the same linear warm-up and cosine decay schedule as the rest of the model.

For Reflective VLA on LIBERO/LIBERO-Plus we use $K{=}8$ context frames, and $K{=}4$ on SimplerEnv-Bridge. Adjacent context frames are separated by one action-chunk stride plus a random gap, sampled from $[0, 15]$ environment steps for LIBERO-family experiments and $[0, 5]$ for SimplerEnv-Bridge; at evaluation the gap is set to zero for deterministic fixed-stride selection.

At evaluation, consecutive context starts are separated by one action-chunk horizon. A historical chunk's consequence observation therefore coincides with the next chunk's starting observation, or with the current query observation for the most recent chunk. The (O, A) variant omits explicit per-triplet consequence entries while retaining the starting observations and the query. During training, positive randomized gaps can make a consequence observation distinct from the next sampled starting observation.

C.2 Evaluation protocol

All models are evaluated from a fixed checkpoint without test-time fine-tuning, using the same pipeline for the reactive baseline and Reflective VLA. For Reflective VLA, the context buffer is reset per episode and populated online from the policy's own executed chunks and reached observations.

LIBERO. We evaluate the four official suites (*Spatial*, *Object*, *Goal*, *Long*) under absolute end-effector control, with 50 rollouts per task (seed 42) and a horizon of 800 steps (900 for *Long*). Success follows the environment's task-completion signal; we report per-suite success and the unweighted average.

LIBERO-Plus and LIBERO-Plus-Hard. For LIBERO-Plus we evaluate the seven standard perturbation categories with one rollout per task; for LIBERO-Plus-Hard we additionally evaluate the multi-camera shift and the robot-calibration shift using the test ranges in Table 4 on the LIBERO base suites. Perturbations are generated deterministically from task and rollout ids, so all methods see identical instances.

SimplerEnv-Bridge. We evaluate the four WidowX tasks (spoon on towel, carrot on plate, cube stacking, eggplant in basket) with 24 rollouts per task. We report per-task success and the unweighted average over the four tasks; with equal rollout counts, this average equals the overall success rate over 96 rollouts. Aggregate rates and absolute gains are rounded to one decimal place after aggregation.

All tables report success rates in percent, averaged over the reported suites or perturbation categories.

C.3 Repeated evaluation and uncertainty

Five-seed LIBERO-Plus evaluation. For each method in Table 6, we independently train five models with different training seeds and evaluate each model at the same scale as the main LIBERO-Plus experiment, using the same benchmark coverage and per-task rollout count (Section C.2). These results are kept separate from Table 2: the means and standard deviations below belong to the same repeated experiment, whose aggregate means differ from the main-table point estimates. Each entry is the mean success rate $\pm$ standard deviation across the five independently trained models and their corresponding evaluations, in percentage points; the standard deviation is neither a standard error nor a 95% confidence interval. The Avg column reports the unweighted seven-category mean and the reported standard deviation of the aggregate across runs; category standard deviations are not averaged to construct it.

Table 6: Additional five-seed evaluation on LIBERO-Plus. Mean $\pm$ standard deviation across five independently trained models per method, each evaluated at the same scale as the main experiment. These statistics correspond to this experiment only; they are not uncertainty estimates for the different point estimates in Table 2.

Method	Cam.	Robot	Lang.	Light	Bg.	Noise	Layout	Avg
Reactive	90.0±2.1	50.0±4.5	94.9±1.3	92.0±1.8	85.8±2.4	90.2±1.9	75.0±3.2	82.6±1.8
Reflective VLA	96.0±1.4	72.9±3.9	95.0±1.2	96.1±1.3	94.1±1.6	91.3±1.7	77.6±2.8	89.0±1.5

Interpretation of variability. The five repetitions are independent training runs, rather than repeated evaluations of a single checkpoint. Their standard deviation describes variation in measured performance across training runs under the stated evaluation protocol. Because each success rate is estimated from a finite set of rollouts, this statistic does not separately isolate training randomness from rollout-level uncertainty. The average gain in this experiment is 6.4 percentage points, compared with 5.4 points in Table 2. On Robot, Reflective VLA reaches $72.9\pm3.9\%$, versus $50.0\pm4.5\%$ for the reactive baseline, a mean gain of 22.9 points. The aggregate means and some category means differ from the main-table point estimates, so the uncertainty estimates are reported only alongside the means of this repeated experiment.

C.4 Inference efficiency and deployment trade-offs

Hardware and timing boundary. Inference timings in Table 7 use a single NVIDIA RTX 4090. The reactive and Reflective VLA policies use bf16 precision, batch size one, and ten flow-matching denoising steps per predicted action chunk. Model latency is measured from feeding the processed image into the model to obtaining the action chunk. The preceding raw-sensor-input-to-model-input pipeline takes approximately 20 ms and is excluded from these model timings. Thus, the reported 124–178 ms values are not end-to-end sensor-to-action latency and do not include physical action execution. We report the mean latency over 100 timed predictions after 10 warm-up predictions. MemoryVLA was timed on the same GPU; its decoder settings and a shared end-to-end timing protocol remain to be documented.

Table 7: Model inference latency on an RTX 4090. Timing ends when the predicted action chunk is available and excludes preprocessing. MemoryVLA is a same-hardware comparison; complete runtime settings remain to be verified.

Policy	Model latency (ms)	Relative latency
Reactive ($K = 1$)	124	1.00×
Reflective VLA ($K = 4$)	148	1.19×
Reflective VLA ($K = 8$)	178	1.43×
MemoryVLA	198	1.60×

Context length and rollout length. With a fixed-size rolling buffer, the retained context does not grow indefinitely with episode duration. KV reuse avoids repeatedly encoding retained history; it does not eliminate attention to that history or its storage. Increasing K therefore increases model latency and memory requirements even though the per-step workload stays bounded as a rollout grows. The latency measurements characterize this cost at the tested context lengths, rather than a constant cost across K.

When is interaction history useful? Calibrated geometry can be sufficient in stable deployments, whereas camera repositioning or changes in the command–response relationship can invalidate a prior calibration. Interaction history offers a way to adapt during task execution when recent actions produce observable evidence about the current deployment. This benefit depends on informative, correctly aligned context; rapidly changing conditions can instead make retained history stale. The camera-shift comparison in Section D.3 evaluates performance with calibrated extrinsics, but does not measure calibration time or establish a universal cost advantage over calibration. For $C = 30$ at the 30 Hz controller rate reported in the real-robot experiment, a chunk spans one second of commanded motion; model latency below this duration alone does not establish uninterrupted control, which also depends on preprocessing, communication, and execution scheduling.

C.5 MORE QUALITATIVE RESULTS

In Figure 6, we show the qualitative results of Reflective VLA on LIBERO-Plus-Hard dataset.

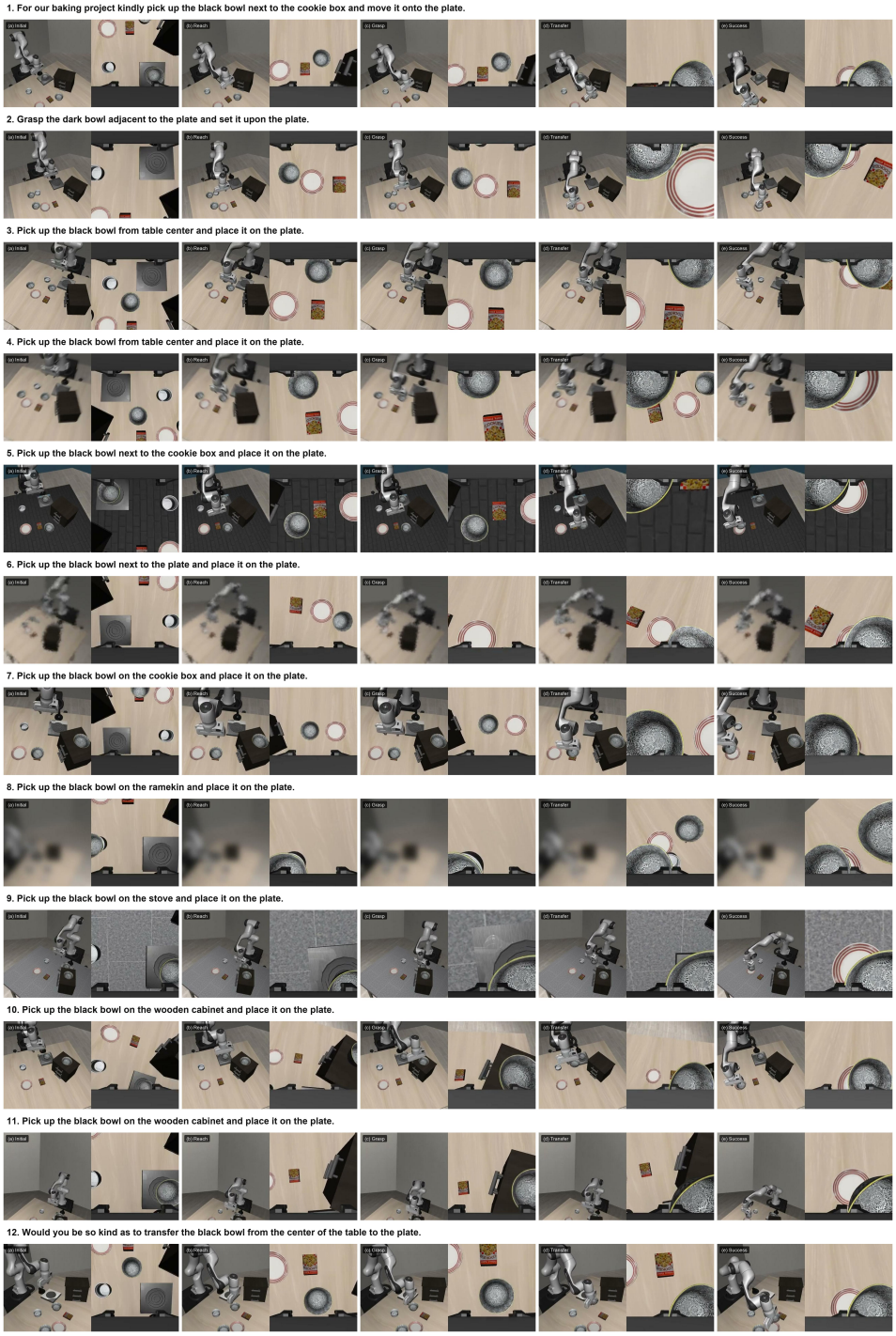


Figure 6: Qualitative results of reflective VLA on LIBERO-Plus-Hard dataset.

D REAL-WORLD PROTOCOL

D.1 HARDWARE AND CONTROL INTERFACE

We use a tabletop setup with an Agilex Piper arm and Intel RealSense D435i cameras. Each policy receives the language instruction, the current third-person view, and the proprioceptive state, and predicts chunked delta end-effector pose and gripper commands with horizon C=30, executed through the same interface for the reactive baseline and Reflective VLA. For Reflective VLA, the context buffer is reset per trial and populated online from executed chunks and their resulting observations; the reactive baseline conditions only on the current observation and instruction. The reactive baseline and Reflective VLA use identical hardware, controller, camera streams, and task initializations, with no test-time fine-tuning or camera-specific calibration.

D.2 TASKS AND CAMERA PLACEMENTS

We use two language-conditioned pick-and-place tasks: a *box* task (placement into a large square box) and a *bowl* task (placement into a smaller bowl, requiring tighter spatial alignment). We use 25 third-person camera placements and collect 500 demonstrations at ten of them. These ten training placements form the Nominal condition. We measure each held-out camera placement's offset from its nearest training placement and normalize it by the standard deviation of the training camera distribution. We denote this dimensionless offset by δ. Small, Medium, and Large each use five held-out placements with $\delta = 0.15$, 0.30, and 0.60, respectively; Nominal placements have $\delta = 0$. Each camera placement is evaluated with five Box and five Bowl rollouts, yielding 250 evaluation rollouts per method in total across the two tasks (125 per task). Thus, Nominal contributes 100 rollouts, and each shifted condition contributes 50. Task semantics and hardware are unchanged. Success is judged by a human (target object inside the container at rollout end).

Table 8: Real-world camera-placement protocol. The normalized offset δ measures deviation from the nearest training camera placement in units of the training camera distribution's standard deviation. Evaluation rollout counts are per method, with five Box and five Bowl rollouts at each placement. The 500 training demonstrations are collected at the ten Nominal placements.

Condition	Normalized offset δ	Placements	Rollouts
Nominal	0	10	100
Small	0.15	5	50
Medium	0.30	5	50
Large	0.60	5	50
Total	—	25	250

D.3 Real-world comparison across camera shifts

We organize the real-world comparison across increasingly shifted camera placements on the Box and Bowl tasks. The methods comprise the reactive baseline, MemoryVLA, the calibration-aware OC-VLA (Zhang et al., 2026), and Reflective VLA. OC-VLA uses calibrated camera extrinsics to express actions in camera coordinates and transform predicted actions back into the robot frame. Reflective VLA instead conditions on interaction history without receiving camera extrinsics.

Table 9: Real-world performance across camera-pose shifts. Success rates (%) averaged over the Box and Bowl tasks. Nominal uses the ten training camera placements; Small, Medium, and Large each use five held-out placements with normalized offsets $\delta = 0.15$, 0.30, and 0.60 from the nearest training placement (Table 8). OC-VLA receives calibrated camera extrinsics.

Method	Nominal	Small shift	Medium shift	Large shift
Reactive	78	62	38	24
MemoryVLA	78	60	40	28
OC-VLA	76	68	58	48
Reflective VLA	**82**	**80**	**76**	**70**

Aggregation. Success rates are averaged equally over Box and Bowl within each condition. Because the two tasks have equal rollout counts, this average equals the pooled success rate over 100 rollouts for Nominal and 50 rollouts for each of Small, Medium, and Large.

OC-VLA improves over the reactive baseline under all three shifted conditions. Reflective VLA further exceeds OC-VLA by 12, 18, and 22 percentage points under Small, Medium, and Large shifts, respectively, showing an advantage for interaction-conditioned adaptation in these evaluated camera configurations.

E Reproducibility and Assets

Assets. Our simulated experiments build on public assets from LIBERO, LIBERO-Plus, BridgeData V2, and SimplerEnv. Pretrained vision-language backbones are obtained from their official releases under their respective licenses. The LIBERO-Plus-Hard perturbations are specified procedurally in our evaluation scripts (Section B). Real-world experiments use the hardware described in Section D.

F Real-World Qualitative Results

Figure 7 groups real-robot placement sequences by object to highlight camera-viewpoint variation within the same task.

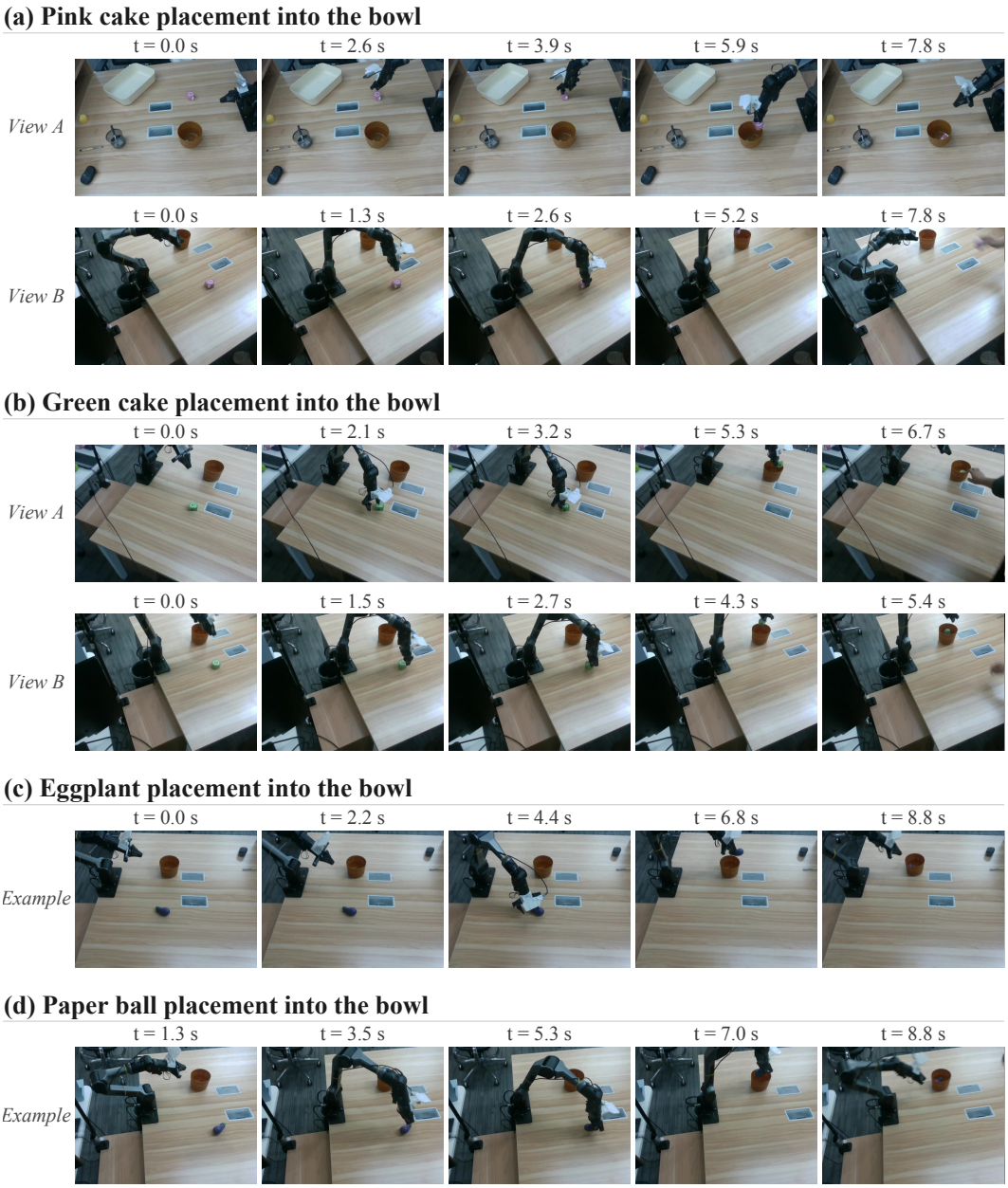


Figure 7: Real-world sequences across camera viewpoints. Third-person recordings of pink and green cake placement, each across two viewpoints, with additional eggplant and paper ball examples. Each row is a contiguous excerpt; columns progress chronologically with clip-relative timestamps. View letters identify examples, not train/test splits. These illustrations do not add quantitative evaluation trials.