LIBERO-PRO: A BENCHMARK FOR EVALUATING THE ROBUSTNESS OF EMBODIED FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

LIBERO has become a de facto standard benchmark for evaluating embodied intelligence. However, high success rates on LIBERO do not necessarily reflect robust embodied capabilities, as models may achieve strong performance by exploiting benchmark-specific patterns. In this work, we introduce **LIBERO-Pro**, a comprehensive benchmark for systematically evaluating the robustness of embodied foundation models. LIBERO-Pro defines 22 static perturbations and 20 runtime interventions, resulting in 1,648 validated perturbation variants. These perturbations cover changes in scene configurations, object properties, visual observations, robot states, temporal conditions, task instructions, and runtime execution states. Each perturbation is carefully designed and undergoes six rounds of human verification to ensure physical plausibility and task validity. We evaluate 11 representative robot policies spanning vision-language-action models, world-action models, and robustness-oriented policies. Our results reveal substantial robustness gaps across all evaluated paradigms: under runtime distribution shifts, the evaluated models suffer an average performance drop of **25.9 percentage points**, with individual drops reaching **32.5 points**. These findings demonstrate that strong nominal benchmark performance does not necessarily translate into robustness under distribution shifts, highlighting the need for robustness-oriented evaluation of embodied foundation models.

1 INTRODUCTION

Reliable evaluation is essential for measuring progress in embodied intelligence (Yu et al., 2020; James et al., 2020; Mees et al., 2022; Gu et al., 2023; Mu et al., 2025; Zhang et al., 2025). LIBERO has emerged as a de facto standard for tabletop manipulation (Liu et al., 2023), with increasingly many systems approaching saturated performance under its standard protocol (Kim et al., 2025b; Black et al., 2025; Kim et al., 2025a; Qu et al., 2025; Lee et al., 2025; Fang et al., 2026). Yet a fundamental question remains: *do these high success rates reflect generalizable embodied capabilities, or merely strong performance within a familiar training–evaluation distribution?*

To examine this question, we revisit high-performing systems on LIBERO and observe three striking failure patterns under simple task variations: (1) when the target object is replaced or even removed, models still execute the original grasping behavior; (2) when the task instruction is replaced with meaningless tokens, models continue to produce nearly identical action trajectories; and (3) when the target object is moved away from its familiar position, models still move toward the location commonly seen during training. These behaviors indicate that strong LIBERO performance can coexist with weak visual grounding, limited sensitivity to instruction semantics, and pronounced reliance on memorized spatial and action patterns.

These failures expose a broader limitation of current evaluation protocols. Standard benchmarks largely test whether a policy can reproduce successful behavior under conditions close to those seen during training (Yu et al., 2020; James et al., 2020; Mees et al., 2022; Gu et al., 2023; Liu et al., 2023), while deployment requires the same system to remain effective when the scene changes before execution, when unexpected events occur during interaction (Li et al., 2024; Fei et al., 2026; Zhang et al., 2026), and when an initial action fails. Consequently, nominal task success alone cannot distinguish memorized competence from robust adaptation. A meaningful evaluation protocol

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.

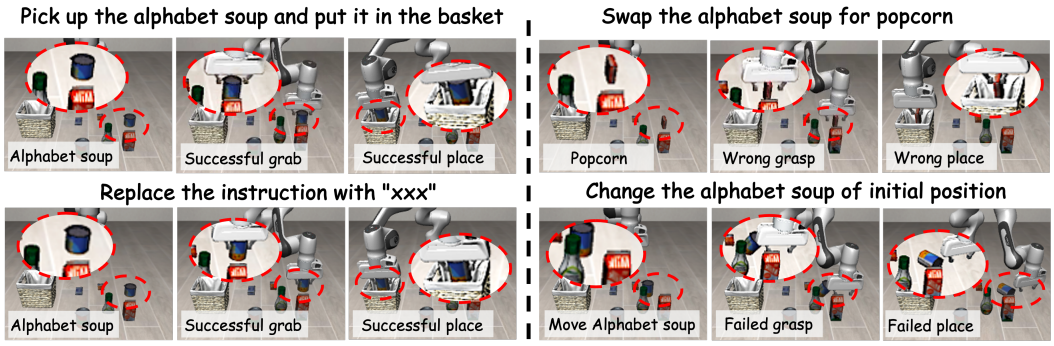

Figure 1: **Failure patterns under task variations.** Examples for the task "*Pick up the alphabet soup and put it in the basket.*" The policy behaves similarly despite changes to the target object, instruction, or object position.

should therefore separate training from evaluation distributions and explicitly test both generalization to static shifts and adaptation to runtime interventions.

To this end, we introduce LIBERO-Pro, a large-scale robustness benchmark for embodied foundation models under static distribution shifts and runtime interventions. LIBERO-Pro defines 42 perturbation types (22 static and 20 runtime), yielding **1,648 perturbed task variants**. These perturbations span scene and object configurations, visual observations, camera settings, robot states, among others. We reserve 8 comprehensively perturbed tasks for evaluation and release the remaining perturbations for training, collecting **39,390 training trajectories**. Each perturbed task undergoes six rounds of human verification to ensure validity and reproducibility. We evaluate 11 representative policies spanning vision-language-action models, world-action models, and robustness-oriented policies. Results reveal substantial robustness degradation under both static and runtime distribution shifts, with runtime shifts causing an average performance drop of **25.9 percentage points**. Moreover, robustness varies considerably across perturbation categories and does not consistently correlate with nominal LIBERO performance, exposing vulnerabilities masked by near-saturated benchmark scores. Our main contributions are threefold:

- We introduce **LIBERO-Pro**, a large-scale robustness benchmark for embodied foundation models, extending LIBERO beyond nominal task success to systematically evaluate robustness under both static distribution shifts and runtime interventions.
- We develop a comprehensive perturbation suite with **42 perturbation types**, yielding **1,648 perturbed task variants** and **39,390 training trajectories** across diverse environments, observations, instructions, robot states, and execution conditions, with rigorous human verification.
- We conduct a large-scale evaluation of **11 representative policies** across different model paradigms, revealing substantial robustness degradation under both static and runtime shifts and showing that nominal LIBERO performance does not consistently reflect robustness.

2 Preliminaries

2.1 Embodied Manipulation Tasks

We consider a language-conditioned manipulation task $\mathcal{T} = (l, e, G)$, where l is the task instruction, e specifies the environment configuration, and G is the task-success predicate. Starting from an initial state s_0, an embodied policy π interacts with the environment for a finite horizon H, receiving an observation o_t and producing an action $a_t \sim \pi(\cdot \mid o_t, l)$ at each step t. The resulting trajectory is denoted by $\tau = (s_0, a_0, \ldots, s_H)$, and an episode is successful if $G(\tau) = 1$.

2.2 Evaluation under Distribution Shifts

The original LIBERO evaluation distribution closely resembles its training distribution. In real-world deployment, however, operating conditions can deviate substantially from those encountered

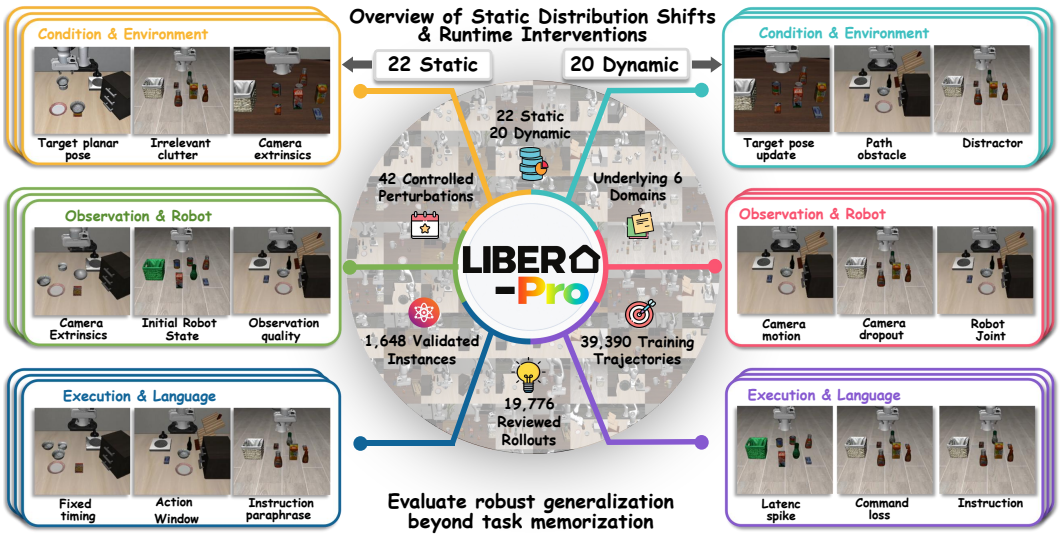

Figure 2: Overview of LIBERO-Pro. LIBERO-Pro extends LIBERO with deployment-relevant variations across environment, observation, robot state, execution, and task instruction dimensions for robustness-oriented evaluation.

during training, making the original evaluation insufficient for assessing policy generalization. We therefore introduce controlled distribution shifts into the LIBERO evaluation setting. Given the original evaluation distribution $\mathcal{D}_{\text{eval}}$, an evaluation transformation ϕ parameterized by η induces a shifted distribution $\mathcal{D}^{\phi}_{\text{eval}} = \phi(\mathcal{D}_{\text{eval}}; \eta)$, while preserving the task objective and success criterion. Performance under $\mathcal{D}^{\phi}_{\text{eval}}$ measures policy generalization when evaluation conditions deviate from those represented during training.

2.3 Static Shifts and Runtime Interventions

We distinguish two types of evaluation transformations according to their temporal behavior during an episode. A *static shift* ϕ^S is applied before policy execution and remains fixed throughout the episode, yielding $\mathcal{T}^S = \phi^S(\mathcal{T}; \eta)$. In contrast, a *runtime intervention* ϕ^D is applied during policy execution, with its occurrence and configuration determined by the interaction history $\tau_{0:t}$, i.e., $\phi^D(\mathcal{T}; t, \tau_{0:t}, \eta)$. Static shifts evaluate generalization under shifted but stationary conditions, whereas runtime interventions evaluate policy robustness to changes that arise during interaction.

3 LIBERO-Pro

3.1 Overview

LIBERO-Pro extends LIBERO with robustness evaluation under relevant variations in environments, observations, robot states, execution processes, and task instructions, while preserving task validity and feasibility. An overview of LIBERO-Pro and its evaluation dimensions is shown in Fig. 2.

3.2 Systematic Distribution Shift Design

We consider two forms of distribution shift according to when the operating condition changes: **Static Distribution Shifts** and **Runtime Distribution Shifts**. Static shifts are introduced before execution and remain fixed throughout an episode, while runtime shifts occur during task execution and alter the environment, observation, robot state, or execution process. The former evaluates out-of-distribution generalization, while the latter evaluates online adaptation and recovery.

Static distribution shifts. Static distribution shifts characterize changes that are established before execution and remain stationary throughout an episode. Let $\phi^S \in \Phi^S$ denote a static shift operator

and η its configuration. Applying ϕ^S to a nominal task $\mathcal{T}$ induces a shifted task $\mathcal{T}^S = \phi^S(\mathcal{T}; \eta)$, where η is specified before the rollout and held fixed throughout execution. LIBERO-Pro instantiates **22 static distribution shifts** over the task environment, observation process, robot configuration, and language condition, covering object configuration and appearance, scene composition, camera and observation conditions, robot initialization and calibration, temporal configuration, and instruction expression. These shifts reflect deployment-relevant variations that arise in real-world robotic manipulation, making robustness to such out-of-distribution conditions essential for deployment.

Runtime distribution shifts. Runtime distribution shifts characterize changes that emerge after task execution has begun. Let $\phi^D \in \Phi^D$ denote a runtime shift operator and η its configuration. Each shift is associated with a trigger function $g_\phi(\tau_{0:t})$ defined over the interaction trajectory $\tau_{0:t}$; the trigger determines when the shift is introduced, while η specifies its realization and temporal extent. LIBERO-Pro instantiates **20 runtime distribution shifts** over the task environment, observation process, robot and action execution, and language condition, covering object or receiver relocation, dynamic occlusion and obstacles, camera and observation failures, frame-stream and synchronization anomalies, transient latency, action-execution disruptions, robot-state changes, instruction updates, and recoverable execution failures. These shifts capture execution-time changes that can occur during real-world deployment, where a robust policy must continuously adapt its behavior and recover from unexpected deviations to maintain stable task execution.

3.3 Data Collection

LIBERO-Pro is built from the 40 tasks in LIBERO-Goal, LIBERO-Spatial, LIBERO-Object, and LIBERO-10. Using random seed 142, we sample two tasks from each suite for held-out evaluation, yielding **8 evaluation tasks** and **32 training tasks**. After applying all applicable distribution shifts and filtering invalid task–shift combinations, the training and evaluation sets contain **1,313** and **335** valid instances, respectively, for a total of **1,648 validated shifted task instances**. For robustness-oriented learning, we collect **30 trajectories for each training instance**, yielding **39,390 trajectories** from $\pi_{0.5}$, GPT-5.6 Sol with Codex, and human demonstrations. All authors conduct **six rounds of iterative human review and correction** over all **1,648 distribution-shift instances**. In each round, every instance is executed twice using $\pi_{0.5}$, producing approximately **3,296 rollout videos** per round and **19,776 reviewed rollouts** across six rounds, totaling more than **82 hours of footage**. Reviewers verify that each shift is correctly realized and that tasks remain semantically valid and physically executable, including spatial consistency after relocation, valid occlusions or obstacles, feasible execution paths, and correct runtime triggering. Temporal and action-stream shifts are additionally checked against execution logs when necessary. Invalid or ambiguous instances are corrected and re-evaluated, while task–shift combinations that fail validation are removed.

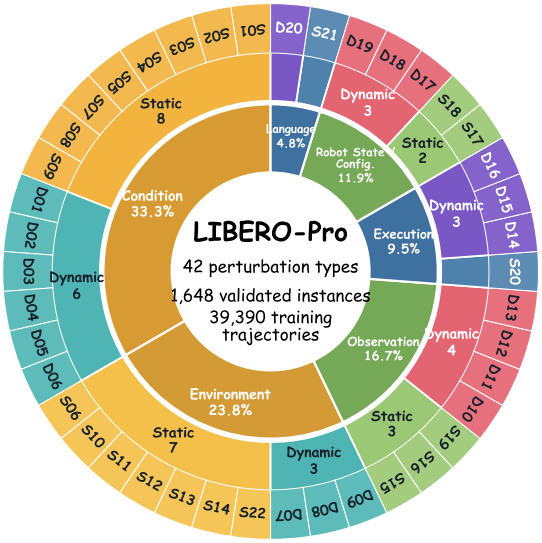

Figure 3: Perturbation distribution in LIBERO-Pro.

4 Experiments

4.1 Experimental Setup

We evaluate 11 representative robot manipulation policies spanning different policy paradigms, using their official checkpoints and inference settings. Experiments are conducted on the 8 held-out

Table 1: Leaderboard of robot-policy robustness under **static distribution shifts**. Results report task success rate (%), averaged across LIBERO-Goal, LIBERO-Object, LIBERO-Spatial, and LIBERO-10 within each perturbation category. Δ denotes the performance change relative to the nominal baseline, i.e., Avg. − Base.

Model	Cond. *S01–S05, S07–S09*	Env. *S06, S10–S14, S22*	Obs. *S15–S16, S19*	Exec. *S20*	Robot *S17–S18*	Lang. *S21*	Avg.	Base	Δ
Vision-Language-Action Policies									
OpenVLA-OFT (Kim et al., 2025a)	44.1	82.0	46.3	0.0	66.5	94.6	58.9	97.1	−38.2
π_0 (Black et al., 2025)	25.0	53.8	60.0	5.0	19.2	46.3	38.6	63.8	−25.1
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	60.7	83.9	43.3	0.8	85.0	93.3	66.7	93.3	−26.6
X-VLA (Zheng et al., 2025)	42.1	81.7	49.3	71.7	61.7	90.8	61.2	95.0	−33.8
GR00T N1.7 (Bjorck et al., 2025)	46.6	79.2	48.5	0.0	77.9	88.8	60.0	90.4	−30.4
MolmoAct2 (Fang et al., 2026)	63.8	85.0	65.1	0.0	85.0	94.6	71.2	97.2	−26.0
World-Action Models									
Cosmos Policy (Kim et al., 2026)	56.9	89.3	70.8	4.2	82.5	93.3	70.8	97.5	−26.7
FastWAM (Yuan et al., 2026)	43.8	84.7	5.2	2.8	87.5	90.3	55.9	97.8	−41.8
LingBot-VA (Li et al., 2026)	52.2	86.1	35.7	86.7	98.3	97.1	68.7	98.5	−29.7
Robustness-Oriented Policies									
Anchor-Align (Dalal et al., 2026)	43.2	74.3	38.6	0.8	76.5	90.8	55.9	91.3	−35.4
RIPT-VLA (Tan et al., 2025)	41.0	74.3	41.8	0.0	63.8	86.7	54.1	87.5	−33.4

Table 2: Leaderboard of robot-policy robustness under **runtime distribution shifts**.

Model	Cond. *D01–D06*	Env. *D07–D09*	Obs. *D10–D13*	Exec. *D14–D16*	Robot *D17–D19*	Lang. *D20*	Avg.	Base	Δ
Vision-Language-Action Policies									
OpenVLA-OFT	42.8	63.1	80.4	85.3	57.6	95.4	64.6	97.1	−32.5
π_0	24.9	56.9	52.7	30.8	7.2	61.3	35.3	63.8	−28.4
$\pi_{0.5}$	60.5	58.1	86.6	88.9	77.5	94.2	73.8	93.3	−19.5
X-VLA	41.7	59.7	93.6	70.1	70.1	95.0	66.0	95.0	−29.0
GR00T N1.7	41.7	66.0	77.9	82.4	60.3	89.6	63.9	90.4	−26.6
MolmoAct2	58.6	74.2	90.3	90.4	79.0	94.2	76.9	97.2	−20.3
World-Action Models									
Cosmos Policy	45.6	84.7	87.9	85.6	70.6	95.8	72.2	97.5	−25.3
FastWAM	43.6	67.1	90.7	89.9	71.4	93.2	70.1	97.8	−27.6
LingBot-VA	77.3	81.5	90.5	96.8	59.3	96.7	81.8	98.5	−16.7
Robustness-Oriented Policies									
Anchor-Align	42.4	47.4	77.4	87.5	58.8	88.3	61.6	91.3	−29.7
RIPT-VLA	38.4	57.6	72.0	76.3	54.6	86.7	58.5	87.5	−29.0

tasks of LIBERO-Pro under nominal execution, 22 static shifts, and 20 runtime shifts. Additional experimental details are provided in Appendix C. Our evaluation is organized around the following research questions:

- **Overall Performance Landscape:** How robust are current robot policies under diverse static and runtime distribution shifts?
- **RQ1:** What limits robustness under runtime distribution shifts?
- **RQ2:** Does robustness compose under compound distribution shifts?
- **RQ3:** Can robustness be learned and generalized to unseen perturbations?

4.2 Overall Performance Landscape

Finding 1: High nominal performance does not imply strong robustness. Table 1 and Table 2 show that, despite strong nominal performance (87.5%–98.5%), all policies degrade substantially under distribution shifts. Static shifts reduce average success rates to 38.6%–71.2% (25.1–41.8 points drop), while runtime shifts reduce them to 35.3%–81.8% (16.7–32.5 points drop). Moreover, robustness rankings do not consistently follow nominal performance. For example, FastWAM achieves higher nominal performance (97.8%) than $\pi_{0.5}$ (93.3%), but its runtime robustness decreases to 70.1%, lower than $\pi_{0.5}$ (73.8%). This ranking reversal indicates that nominal task success

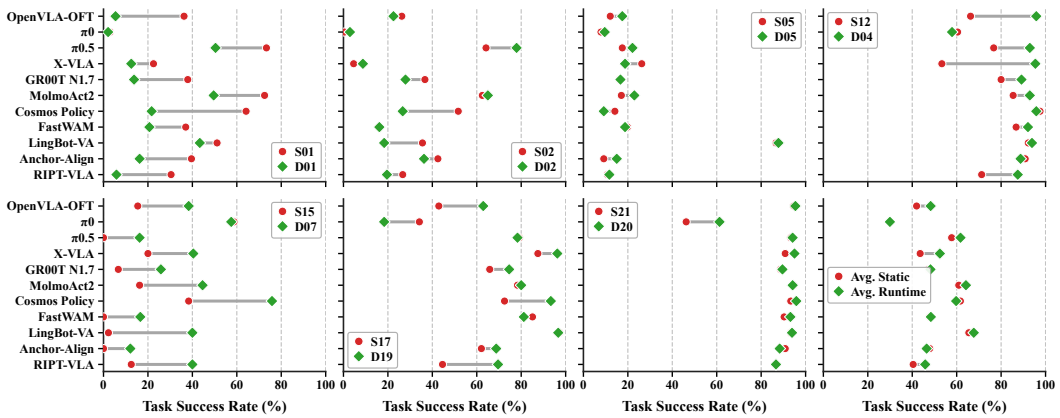

Figure 4: Comparison between matched static and runtime perturbations. Each pair contains a static shift occurring before execution and its runtime counterpart triggered during execution, including object pose changes (S01/D01, S02/D02), path constraints (S05/D05), occlusion (S12/D04), camera changes (S15/D07), robot-state variations (S17/D19), and language variations (S21/D20).

does not necessarily predict robustness under runtime distribution shifts. Therefore, robustness evaluation reveals additional capabilities beyond standard task solving metrics.

Finding 2: Runtime robustness requires adaptation beyond static generalization. Figure 4 compares semantically matched static–runtime perturbation pairs, including object relocation, goal relocation, path obstruction, occlusion, camera changes, and language variation. Despite similar perturbation mechanisms, their effects differ across static and runtime settings. For example, LingBot-VA improves from 68.7% under static shifts to 81.8% under runtime shifts, indicating that runtime robustness depends on closed-loop adaptation rather than static generalization alone.

Finding 3: Different perturbation categories expose heterogeneous robustness bottlenecks. Robustness varies substantially across perturbation dimensions. Language variations remain relatively easy, with all models achieving over 46% success under S21, whereas execution-related shifts are considerably more challenging, reducing OpenVLA-OFT and π_0 to 0.0% and 5.0% under S20. Runtime shifts further expose distinct weaknesses in observation, execution, and physical-state adaptation, showing that robustness is inherently multidimensional.

4.3 RQ1: What Limits Robustness under Runtime Distribution Shifts?

Table 3: Feedback utilization gain G_{feedback} (pp) across dynamic perturbation categories.

Model	Cond. *D01–D06*	Env. *D07–D09*	Obs. *D10–D13*	Exec. *D14–D16*	Robot *D17–D19*	Lang. *D20*	Avg.
OpenVLA-OFT	-26.6	-22.8	-15.0	-8.7	-27.3	-1.1	-19.9
$\pi_{0.5}$	-3.5	-19.2	-3.2	-5.3	+7.5	+2.1	-4.1
Anchor-Align	-15.1	-19.5	-12.9	-10.8	-8.9	-7.9	-13.4
FastWAM	-15.6	-9.2	-1.4	-7.5	-6.8	-3.3	-8.7
Cosmos Policy	-11.8	-10.3	-6.5	-13.2	+1.1	-5.4	-8.5

We investigate why policies fail under runtime distribution shifts through three controlled interventions targeting feedback utilization, closed-loop update frequency, and post-perturbation recovery.

RQ1-A: Can Policies Use Updated Observations? We examine whether policies can exploit observations updated after runtime perturbations. Under *live* feedback, the policy continuously receives post-perturbation visual observations, whereas under *freeze* feedback, it repeatedly receives the last observation before the perturbation and therefore cannot observe the change. Here, S_1 denotes the success rate under a dynamic perturbation, while S_0 denotes the success rate under the corresponding no-perturbation condition. Therefore, $S_{1,\text{live}}$ and $S_{1,\text{freeze}}$ measure performance with

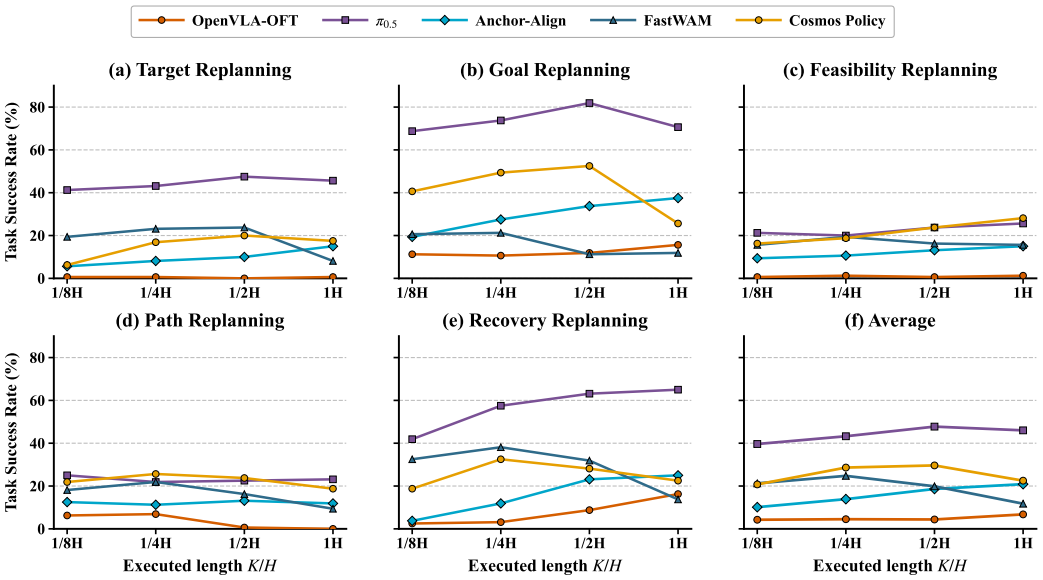

Figure 5: Effect of policy execution horizon on robustness to dynamic perturbations. Panels (a)–(e) report task success rates under five representative perturbation types; panel (f) shows the average across these perturbations.

and without access to updated observations after a perturbation, whereas $S_{0,\text{live}}$ and $S_{0,\text{freeze}}$ capture the normal feedback effect in the absence of perturbations. We define

$$G_{\text{feedback}} = (S_{1,\text{live}} - S_{1,\text{freeze}}) - (S_{0,\text{live}} - S_{0,\text{freeze}}),$$

where positive values indicate that updated observations provide additional benefit beyond the policy's normal feedback response. Table 3 summarizes the results, with complete results reported in Appendix A.2. Table 3 reveals two main patterns. Table 3 reveals two main patterns. **(1) Limited utilization of updated observations.** Updated observations do not consistently improve adaptation. All policies exhibit negative feedback gains across most perturbation categories, indicating that observing environmental changes alone is insufficient to trigger effective corrective behaviors. For example, OpenVLA-OFT achieves an overall gain of -19.9 pp, suggesting that the main bottleneck lies in converting new observations into appropriate actions rather than observation availability itself. **(2) Feedback utilization depends on perturbation type.** While most categories show limited gains, $\pi_{0.5}$ and Cosmos Policy obtain positive gains on robot-state changes (D17–D19), with $+7.5$ pp and $+1.1$ pp, respectively. In contrast, environment and observation shifts consistently lead to negative gains, indicating that these perturbations require changes beyond state tracking, such as updating perception or action strategies.

RQ1-B: Does Closed-Loop Update Frequency Limit Robustness? We study whether action chunking delays the use of updated observations by fixing the prediction horizon H and varying the number of executed actions K before re-observation. We evaluate target replanning (D01), goal replanning (D02), feasibility replanning (D03), path replanning (D05), and recovery replanning (D18) under execution ratios K/H. Figure 5 presents the results. **(1) Replanning frequency exhibits a non-monotonic effect.** Average success increases from 21.6% at $K/H = 1$ to 24.0% at $1/2$, but drops to 19.2% at $1/8$, suggesting that sparse updates delay adaptation while overly frequent replanning can disrupt execution consistency. **(2) Frequency sensitivity is policy-dependent.** FastWAM and Cosmos Policy benefit more from shorter execution intervals, whereas Anchor-Align performs better with longer horizons. OpenVLA-OFT remains weak across settings, indicating that update frequency alone cannot compensate for limited perturbation handling.

Table 4: Average performance and intervention gains in RQ1-C. Δ denotes the change in success rate relative to normal continuation.

Model	Continue	Δ_{Hold}	Δ_{Retry}	Δ_{Reset}
OpenVLA-OFT	64.6	-15.7	-40.8	-31.6
$\pi_{0.5}$	73.8	+18.9	-24.2	+0.7
Anchor-Align	61.7	+28.1	-16.8	+1.0
FastWAM	70.1	+24.4	-20.9	-0.4
Cosmos Policy	72.2	+26.0	-23.0	+2.5

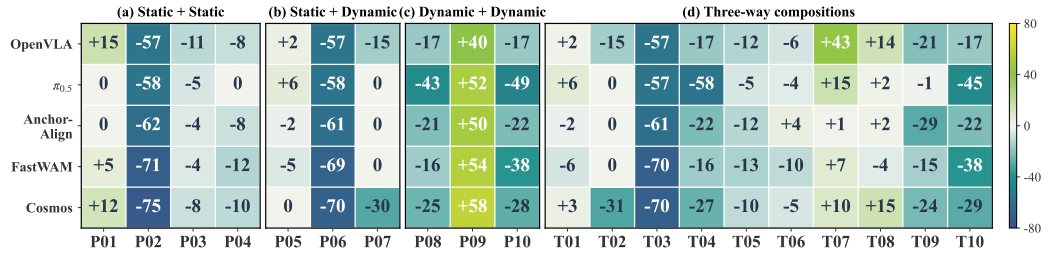

Figure 6: Weakest-single gap $\Delta_m(\mathcal{C})$ under composed perturbations. The heatmap compares static–static, static–dynamic, dynamic–dynamic, and three-way compositions across policies.

RQ1-C: What Prevents Recovery after a Runtime Shift? We study recovery after runtime shifts using three post-perturbation interventions: *Hold* pauses execution until the perturbation ends, *Retry* replays actions from the unperturbed trajectory, and *Reset* restores the robot to its pre-perturbation pose. For each intervention I, we report the relative gain $\Delta_I = S_I - S_{\text{continue}}$. Table 4 summarizes the average effects, with detailed results reported in Appendix A.3.

From Table 4, we make three observations. **(1) Continued execution can amplify runtime failures.** *Hold* provides the most consistent improvement, increasing average success by 18.9–28.1 points for four of five policies. OpenVLA-OFT instead drops by 15.7 points, showing that pausing alone cannot compensate for limited adaptation. **(2) Replaying actions does not enable recovery.** *Retry* reduces average success by 16.8–40.8 points across all policies, indicating that previously valid actions often become unsuitable after the task state changes and must be replanned. **(3) Robot-state restoration provides limited benefit.** *Reset* yields small changes for $\pi_{0.5}$, Anchor-Align, FastWAM, and Cosmos Policy, while OpenVLA-OFT drops by 31.6 points. Post-shift failures stem more from changed task states and invalid plans than robot pose deviation alone.

4.4 RQ2: Does Robustness Compose?

Real-world perturbations often occur jointly, raising the question of whether robustness to individual shifts transfers to compound settings. For a model m and perturbation composition $\mathcal{C} = \{c_1, \ldots, c_k\}$, we define the weakest-single gap as $\Delta_m(\mathcal{C}) = p_m(\mathcal{C}) - \min_{c_i \in \mathcal{C}} p_m(c_i)$. A large negative value indicates additional degradation introduced by composition. Figure 6 summarizes these gaps, while Appendix 13 reports detailed success rates.

Table 5: Key evidence for non-compositional robustness. Although all models perform reasonably well on the constituent perturbations S09, S13, D13, and S22, they all collapse on the composed settings P02, P06, and T03.

Model	S09	S13	D13	S22	P02: S09+S13	P06: S09+D13	T03: S09+S22+D13
OpenVLA-OFT	69.6	94.2	69.6	93.3	**12.5**	**12.5**	**12.5**
$\pi_{0.5}$	70.0	92.5	83.8	94.2	**12.5**	**12.1**	**12.9**
Anchor-Align	72.5	86.7	79.6	87.9	**10.0**	**11.3**	**11.7**
FastWAM	80.3	91.2	83.7	88.3	**9.2**	**11.7**	**10.0**
Cosmos Policy	87.5	99.2	81.7	96.7	**12.9**	**11.7**	**12.1**

Robustness to individual perturbations does not reliably compose. Table 5 shows that P02 (`S09+S13`), P06 (`S09+D13`), and T03 (`S09+S22+D13`) reduce success to only 9.2%–12.9%, although their constituent perturbations remain individually manageable (69.6%–99.2%). Thus, robustness to isolated shifts can collapse when multiple assumptions are violated jointly.

Recovery capability is also non-compositional. As shown in Appendix A.4, $\pi_{0.5}$ performs well on D02, D18, and D20 individually, yet drops to 22.9% on P08 (`D02+D18`) and 17.5% on P10 (`D18+D20`). This indicates that recovery from one runtime disturbance does not guarantee recovery under concurrent shifts, and that composition failures depend more on interactions between perturbation mechanisms than on perturbation count alone.

Table 6: Transfer of robustness from training set to unseen tasks. Each column reports the average success rate over a static–dynamic perturbation pair. *Gain* denotes the improvement of Ours over $\pi_{0.5}$.

Model	Object Pose	Receiver Pose	Path Obstacle	Camera Calibration	Robot State	Total
OpenVLA-OFT	20.8	24.4	14.8	26.9	52.9	28.0
$\pi_{0.5}$	61.9	71.0	19.8	8.1	78.7	47.9
Anchor-Align	27.9	39.4	12.1	6.0	65.4	30.2
FastWAM	28.8	16.0	19.3	8.3	83.1	31.1
Cosmos Policy	42.9	39.2	11.7	**57.1**	82.9	46.7
Ours	**68.8**	**73.2**	**96.3**	20.8	**91.7**	**70.1**
Gain	+6.9	+2.1	+76.5	+12.7	+12.9	+22.2

4.5 RQ3: Can Robustness Be Learned and Generalized?

We ask whether robustness can be learned from limited demonstrations and transferred to unseen tasks. We train $\pi_{0.5}$ on eight training tasks with ten trajectories per task, while keeping all evaluation tasks disjoint. We consider five semantically matched static–dynamic perturbation families: S01/D01 (object pose), S02/D02 (receiver pose), S05/D05 (path obstacle), S15/D07 (camera calibration), and S17/D19 (robot state).

Robustness can be learned from limited demonstrations. With only 80 training trajectories for each perturbation, our model achieves 70.1% average robustness on unseen tasks, outperforming existing policies (28.0%–47.9%) in Table 6. The largest gain appears on path changes (S05/D05), where our model reaches 96.3%, compared with below 20% for all baselines.

Generalization depends on perturbation semantics. Learned robustness transfers well in receiver pose (73.2%) and robot-state shifts (91.7%), but remains limited under camera calibration shifts (20.8%). This suggests robustness generalizes better when perturbation semantics preserve stable task constraints than when they require broader perception–action recalibration.

5 Related Work

Robot manipulation benchmarks such as Meta-World, RLBench, CALVIN, ManiSkill2, and LIBERO have enabled standardized evaluation of multi-task and language-conditioned manipulation (Yu et al., 2020; James et al., 2020; Mees et al., 2022; Gu et al., 2023; Liu et al., 2023), while recent benchmarks further expand task, scene, and embodiment diversity (Nasiriany et al., 2024; Mu et al., 2025; Zhang et al., 2025). Meanwhile, large-scale robot data and foundation policies have substantially improved cross-task and cross-environment generalization (Open X-Embodiment Collaboration et al., 2023; Khazatsky et al., 2024; Kim et al., 2025b; Black et al., 2025; Kim et al., 2025a), with world-action models further incorporating predictive world representations into control (Kim et al., 2026; Li et al., 2026; Yuan et al., 2026). Recent studies have begun to examine robustness beyond nominal task success through domain randomization and controlled perturbations (Li et al., 2024; Chen et al., 2025; Zhang et al., 2026). In particular, LIBERO-Plus evaluates VLA robustness under progressively challenging static distribution shifts (Fei et al., 2026), but does not consider perturbations that occur during task execution. LIBERO-Pro extends robustness evaluation to both static distribution shifts and runtime interventions, further examining closed-loop adaptation, recovery, compound perturbations, and robustness transfer.

6 Conclusion

We present LIBERO-Pro, a systematic benchmark for evaluating the robustness of embodied foundation models under static distribution shifts and runtime interventions. Our evaluation reveals substantial robustness gaps that remain hidden by strong nominal performance, while further analyses highlight challenges in online adaptation, recovery, compound robustness, and robustness transfer. We hope LIBERO-Pro provides a practical foundation for developing embodied policies that remain reliable under diverse and evolving deployment conditions.

AI USE STATEMENT

Generative AI tools were used to assist with code development and debugging, training trajectory collection, and language editing of the manuscript. All AI-assisted code, trajectories, experimental outputs, and textual revisions were carefully reviewed and verified by the authors. The research design, methodology, experimental analysis, and scientific conclusions were determined by the authors, who take full responsibility for all content, results, and claims presented in this work.

ETHICS STATEMENT

This work does not involve human subjects or personally identifiable information. All benchmark instances and collected trajectories were carefully reviewed and validated by the authors to ensure task validity, physical feasibility, and experimental integrity. LIBERO-Pro is intended to support the development and evaluation of more reliable embodied systems under distribution shifts and unexpected runtime conditions. The authors take responsibility for the appropriate collection, validation, and reporting of all data and experimental results presented in this work.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the LIBERO-Pro benchmark construction, distribution-shift design, data collection, and evaluation protocol in the main paper. Additional implementation details and complete experimental results are provided in the Appendix. All evaluated policies use their official checkpoints and inference settings under consistent evaluation protocols. The supplementary materials include the LIBERO-Pro benchmark, collected training trajectories, evaluation code, and configuration files to facilitate reproduction of our experiments and future research.

REFERENCES

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

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. π_0: A vision-language-action flow model for general robot control. In *Robotics: Science and Systems*, 2025.

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

Dwip Dalal, Shivansh Patel, Chahit Jain, Jeonghwan Kim, Utkarsh Mishra, Alex Baratian, Hyeonjeong Ha, Heng Ji, Svetlana Lazebnik, and Unnat Jain. Generalizable vla finetuning via representation anchoring and language-action alignment. *arXiv preprint arXiv:2607.13429*, 2026.

Haoquan Fang, Jiafei Duan, Donovan Clay, Sam Wang, Shuo Liu, Weikai Huang, Xiang Fan, Wei-Chuan Tsai, Shirui Chen, Yi Ru Wang, Shanli Xing, Jaemin Cho, Jae Sung Park, Ainaz Eftekhar, Peter Sushko, Karen Farley, Angad Wadhwa, Cole Harrison, Winson Han, Ying-Chun Lee, Eli VanderBilt, Rose Hendrix, Suveen Ellawela, Lucas Ngoo, Joyce Chai, Zhongzheng Ren, Ali Farhadi, Dieter Fox, and Ranjay Krishna. Molmoact2: Action reasoning models for real-world deployment. *arXiv preprint arXiv:2605.02881*, 2026.

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*, pp. 38574–38583, 2026.

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.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. In *Proceedings of Robotics: Science and Systems*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025a.

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 *Proceedings of the 8th Conference on Robot Learning*, volume 270, pp. 2679–2713, 2025b.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos policy: Fine-tuning video models for visuomotor control and planning. In *International Conference on Learning Representations*, 2026.

Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, Winson Han, Wilbert Pumacay, Angelica Wu, Rose Hendrix, Karen Farley, Eli VanderBilt, Ali Farhadi, Dieter Fox, and Ranjay Krishna. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, Yujun Shen, and Yinghao Xu. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

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*, 2024.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

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

Yao Mu, Tianxing Chen, Zanxin Chen, Shijia Peng, Zhiqian Lan, Zeyu Gao, Zhixuan Liang, Qiaojun Yu, Yude Zou, Mingkun Xu, Lunkai Lin, Zhiqiang Xie, Mingyu Ding, and Ping Luo. Robotwin: Dual-arm robot benchmark with generative digital twins. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27649–27660, 2025.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems*, 2024.

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, and Xuelong Li. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models. *arXiv preprint arXiv:2505.17016*, 2025.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of the Conference on Robot Learning*, pp. 1094–1100, 2020.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

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*, pp. 11142–11152, 2025.

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. *arXiv preprint arXiv:2603.22078*, 2026.

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. *arXiv preprint arXiv:2510.10274*, 2025.

A Additional Experimental Results

This appendix provides the complete LIBERO-Pro results supplementing the feedback-utilization, recovery-intervention, and robustness-composition analyses in the main paper.

A.1 Main Results

Table 7: Static and runtime perturbation results on LIBERO-Goal. All values are reported in percentages (%).

(a) Static Perturbations

Model	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22
OpenVLA-OFT	33.3	0.0	0.0	0.0	0.0	86.7	58.3	83.3	81.7	91.7	85.0	55.0	90.0	93.3	0.0	8.3	0.0	81.7	90.0	0.0	88.3	88.3
π_0	3.3	0.0	0.0	0.0	0.0	48.3	51.7	58.3	56.7	55.0	45.0	63.3	65.0	43.3	50.0	51.7	30.0	1.7	51.7	13.3	40.0	51.7
$\pi_{0.5}$	73.3	51.7	15.0	61.7	6.7	95.0	75.0	93.3	96.7	93.3	95.0	66.7	96.7	95.0	0.0	0.0	50.0	93.3	95.0	0.0	96.7	98.3
X-VLA	23.3	0.0	0.0	0.0	0.0	93.3	93.3	96.7	100.0	95.0	85.0	46.7	93.3	96.7	0.0	0.0	65.0	28.3	98.3	48.3	95.0	95.0
GR00T N1.7	21.7	1.7	30.0	0.0	0.0	73.3	75.0	85.0	81.7	86.7	73.3	71.7	83.3	80.0	0.0	0.0	30.0	85.0	90.0	0.0	81.7	83.3
MolmoAct2	73.3	46.7	6.7	28.3	3.3	86.7	66.7	100.0	96.7	98.3	61.7	86.7	96.7	96.7	1.7	55.0	58.3	96.7	98.3	0.0	93.3	95.0
Anchor-Align	35.0	5.0	0.0	5.0	0.0	61.7	50.0	90.0	91.7	80.0	56.7	85.0	83.3	83.3	0.0	0.0	21.7	91.7	91.7	0.0	91.7	90.0
RIPT-VLA	11.7	0.0	0.0	0.0	0.0	56.7	51.7	53.3	51.7	56.7	58.3	58.3	56.7	56.7	0.0	0.0	0.0	53.3	56.7	0.0	56.7	46.7
Cosmos Policy	56.7	43.3	23.3	13.3	23.3	73.3	90.0	80.0	90.0	96.7	83.3	96.7	100.0	96.7	23.3	53.3	56.7	96.7	100.0	16.7	86.7	96.7
FastWAM	43.3	0.0	0.0	1.7	0.0	85.0	55.0	81.7	91.7	90.0	86.7	76.7	76.7	76.7	0.0	0.0	63.3	90.0	0.0	0.0	76.7	83.3
LingBot-VA	38.3	5.0	45.0	3.3	78.3	58.3	46.7	88.3	48.3	86.7	90.0	90.0	88.3	93.3	0.0	0.0	96.7	95.0	66.7	73.3	90.0	91.7

(b) Runtime Perturbations

Model	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20
OpenVLA-OFT	8.3	0.0	50.0	88.3	30.0	85.0	8.3	25.0	85.0	71.7	68.3	60.0	50.0	38.3	86.7	86.7	86.7	1.7	18.3	91.7
π_0	5.0	0.0	40.0	51.7	13.3	50.0	58.3	43.3	55.0	65.0	56.7	56.7	30.0	15.0	33.3	20.0	3.3	0.0	26.7	55.0
$\pi_{0.5}$	38.3	90.0	50.0	100.0	40.0	96.7	0.0	50.0	98.3	86.7	86.7	86.7	85.0	88.3	93.3	93.3	93.3	78.3	56.7	93.3
X-VLA	33.3	0.0	50.0	93.3	28.3	88.3	0.0	58.3	96.7	95.0	93.3	96.7	71.7	91.7	85.0	3.3	85.0	5.0	96.7	93.3
GR00T N1.7	23.3	20.0	50.0	75.0	36.7	65.0	0.0	63.3	88.3	65.0	46.7	55.0	55.0	56.7	76.7	80.0	71.7	13.3	48.3	81.7
MolmoAct2	35.0	76.7	50.0	96.7	41.7	96.7	35.0	83.3	96.7	98.3	93.3	93.3	90.0	93.3	93.3	96.7	93.3	68.3	51.7	93.3
Anchor-Align	31.7	11.7	50.0	91.7	36.7	41.7	0.0	5.0	93.3	70.0	63.3	73.3	78.3	66.7	96.7	86.7	76.7	25.0	38.3	90.0
RIPT-VLA	0.0	0.0	48.3	56.7	21.7	51.7	3.3	5.0	56.7	50.0	53.3	46.7	30.0	30.0	55.0	43.3	46.7	1.7	28.3	56.7
Cosmos Policy	10.0	23.3	36.7	96.7	13.3	93.3	56.7	73.3	96.7	100.0	90.0	90.0	80.0	73.3	96.7	93.3	90.0	10.0	93.3	90.0
FastWAM	31.7	1.7	48.3	83.3	28.3	85.0	0.0	81.7	88.3	91.7	88.3	80.0	78.3	81.7	91.7	80.0	91.7	38.3	48.3	86.7
LingBot-VA	40.0	0.0	93.3	91.7	76.7	91.7	0.0	63.3	90.0	81.7	76.7	60.0	78.3	95.0	93.3	88.3	85.0	1.7	96.7	90.0

Table 8: Static and runtime perturbation results on LIBERO-Object. All values are reported in percentages (%).

(a) Static Perturbations

Model	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22
OpenVLA-OFT	8.3	1.7	0.0	16.7	1.7	0.0	95.0	91.7	60.0	98.3	80.0	70.0	98.3	100.0	10.0	20.0	51.7	100.0	96.7	0.0	96.7	100.0
π_0	1.7	0.0	0.0	0.0	0.0	0.0	61.7	78.3	81.7	81.7	41.7	71.7	88.3	80.0	85.0	76.7	20.0	0.0	73.3	1.7	41.7	85.0
$\pi_{0.5}$	88.3	85.0	21.7	56.7	3.3	0.0	71.7	61.7	51.7	96.7	75.0	95.0	98.3	98.3	0.0	3.3	98.3	100.0	100.0	0.0	100.0	100.0
X-VLA	10.0	0.0	0.0	5.0	18.3	0.0	78.3	88.3	100.0	98.3	65.0	80.0	100.0	98.3	0.0	18.3	96.7	25.0	98.3	91.7	100.0	100.0
GR00T N1.7	30.0	45.0	5.0	0.0	3.3	0.0	66.7	93.3	95.0	96.7	68.3	98.3	100.0	95.0	0.0	0.0	86.7	100.0	98.3	0.0	98.3	96.7
MolmoAct2	100.0	80.0	5.0	96.7	5.0	0.0	76.7	100.0	100.0	100.0	85.0	100.0	100.0	100.0	0.0	100.0	98.3	96.7	98.3	0.0	100.0	100.0
Anchor-Align	1.7	50.0	0.0	0.0	0.0	0.0	58.3	100.0	96.7	33.3	68.3	96.7	91.7	100.0	0.0	0.0	95.0	100.0	96.7	0.0	96.7	100.0
RIPT-VLA	1.7	0.0	0.0	18.3	0.0	0.0	91.7	96.7	50.0	98.3	81.7	88.3	98.3	98.3	5.0	5.0	51.7	98.3	98.3	0.0	98.3	95.0
Cosmos Policy	50.0	36.7	20.0	3.3	0.0	20.0	100.0	100.0	96.7	100.0	86.7	100.0	100.0	100.0	20.0	66.7	66.7	86.7	96.7	0.0	100.0	100.0
FastWAM	39.7	7.7	0.0	1.7	0.0	0.0	81.7	99.0	100.0	100.0	81.7	100.0	100.0	100.0	0.0	0.0	98.3	100.0	38.0	0.0	100.0	100.0
LingBot-VA	73.3	95.0	100.0	15.0	100.0	75.0	40.0	100.0	0.0	100.0	98.3	100.0	96.7	100.0	5.0	31.7	100.0	100.0	100.0	88.3	100.0	98.3

(b) Runtime Perturbations

Model	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20
OpenVLA-OFT	0.0	1.7	0.0	98.3	0.0	100.0	0.0	76.7	98.3	100.0	98.3	95.0	91.7	80.0	95.0	100.0	91.7	50.0	90.0	95.0
π_0	0.0	0.0	0.0	78.3	6.7	75.0	80.0	80.0	73.3	76.7	88.3	81.7	33.3	20.0	73.3	58.3	1.7	0.0	3.3	75.0
$\pi_{0.5}$	81.7	86.7	0.0	98.3	0.0	96.7	0.0	98.3	96.7	91.7	93.3	91.7	85.0	88.3	96.7	100.0	90.0	63.3	88.3	100.0
X-VLA	1.7	0.0	0.0	100.0	3.3	95.0	13.3	1.7	100.0	98.3	96.7	100.0	93.3	98.3	95.0	58.3	88.3	33.3	100.0	100.0
GR00T N1.7	15.0	15.0	0.0	100.0	3.3	100.0	0.0	90.0	100.0	96.7	91.7	85.0	85.0	71.7	95.0	91.7	100.0	45.0	100.0	98.3
MolmoAct2	91.7	78.3	0.0	100.0	3.3	100.0	20.0	93.3	98.3	100.0	100.0	95.0	93.3	96.7	100.0	100.0	100.0	88.3	95.0	100.0
Anchor-Align	3.3	38.3	0.0	96.7	0.0	100.0	0.0	41.7	95.0	73.3	71.7	78.3	71.7	88.3	98.3	100.0	100.0	21.7	83.3	96.7
RIPT-VLA	8.3	5.0	0.0	96.7	0.0	100.0	10.0	70.0	95.0	95.0	93.3	85.0	63.3	76.7	100.0	98.3	93.3	40.0	93.3	100.0
Cosmos Policy	26.7	10.0	0.0	100.0	3.3	100.0	80.0	100.0	100.0	83.3	96.7	93.3	90.0	90.0	93.3	100.0	93.3	73.3	100.0	100.0
FastWAM	37.3	4.0	0.0	98.3	1.7	100.0	0.0	99.0	100.0	100.0	100.0	100.0	99.0	99.0	100.0	100.0	100.0	59.0	100.0	98.3
LingBot-VA	78.3	0.0	100.0	100.0	100.0	100.0	83.3	98.3	100.0	100.0	100.0	100.0	98.3	100.0	100.0	98.3	98.3	0.0	100.0	100.0

Table 9: Static and runtime perturbation results on LIBERO-Spatial. All values are reported in percentages (%).

(a) Static Perturbations

Model	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22
OpenVLA-OFT	73.3	76.7	30.0	N/A	0.0	80.0	100.0	93.3	98.3	88.3	100.0	98.3	93.3	98.3	51.7	81.7	96.7	86.7	100.0	0.0	98.3	100.0
π_0	6.7	0.0	8.3	N/A	0.0	40.0	61.7	71.7	65.0	55.0	66.7	61.7	56.7	60.0	58.3	71.7	55.0	10.0	68.3	3.3	60.0	60.0
$\pi_{0.5}$	95.0	100.0	40.0	N/A	6.7	85.0	100.0	98.3	93.3	96.7	100.0	81.7	100.0	100.0	0.0	100.0	98.3	100.0	100.0	0.0	98.3	98.3
X-VLA	40.0	1.7	3.3	N/A	0.0	63.3	98.3	95.0	93.3	81.7	90.0	31.7	88.3	98.3	31.7	48.3	93.3	65.0	95.0	98.3	93.3	91.7
GR00T N1.7	85.0	76.7	23.3	N/A	0.0	81.7	100.0	95.0	95.0	95.0	93.3	98.3	100.0	100.0	23.3	98.3	96.7	100.0	98.3	0.0	98.3	100.0
MolmoAct2	75.0	95.0	31.7	N/A	0.0	85.0	100.0	88.3	100.0	95.0	100.0	98.3	98.3	100.0	40.0	98.3	96.7	100.0	100.0	0.0	96.7	98.3
Anchor-Align	95.0	95.0	36.7	N/A	0.0	21.7	100.0	68.3	81.7	91.7	98.3	98.3	93.3	96.7	0.0	95.0	100.0	100.0	98.3	0.0	98.3	100.0
RIPT-VLA	78.3	83.3	28.3	N/A	0.0	71.7	98.3	95.0	96.7	88.3	95.0	95.0	93.3	96.7	45.0	85.0	100.0	91.7	100.0	0.0	100.0	100.0
Cosmos Policy	73.3	56.7	53.3	N/A	13.3	90.0	100.0	93.3	96.7	90.0	90.0	93.3	96.7	96.7	63.3	93.3	83.3	93.3	90.0	0.0	90.0	100.0
FastWAM	40.0	49.0	27.0	N/A	1.0	81.0	98.0	95.0	91.0	88.0	91.0	77.0	88.0	94.0	0.0	23.0	92.0	98.0	1.0	11.0	88.0	85.0
LingBot-VA	83.3	75.0	30.0	N/A	93.3	0.0	0.0	91.7	0.0	98.3	100.0	93.3	100.0	98.3	13.3	21.7	100.0	100.0	96.7	93.3	100.0	100.0

(b) Runtime Perturbations

Model	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20
OpenVLA-OFT	13.3	70.0	1.7	100.0	0.0	96.7	100.0	60.0	100.0	93.3	98.3	71.7	38.3	76.7	93.3	95.0	98.3	13.3	96.7	100.0
π_0	3.3	1.7	15.0	60.0	0.0	48.3	50.0	60.0	56.7	58.3	55.0	48.3	31.7	16.7	25.0	18.3	3.3	0.0	28.3	66.7
$\pi_{0.5}$	76.7	98.3	26.7	98.3	5.0	96.7	21.7	95.0	100.0	95.0	98.3	95.0	93.3	88.3	95.0	98.3	100.0	95.0	98.3	100.0
X-VLA	15.0	18.3	0.0	90.0	0.0	93.3	86.7	78.3	95.0	93.3	93.3	96.7	86.7	93.3	81.7	28.3	76.7	25.0	93.3	91.7
GR00T N1.7	16.7	51.7	1.7	100.0	0.0	88.3	78.3	100.0	100.0	98.3	98.3	88.3	76.7	90.0	88.3	96.7	83.3	38.3	100.0	100.0
MolmoAct2	70.0	83.3	26.7	90.0	1.7	93.3	56.7	100.0	100.0	100.0	98.3	98.3	86.7	90.0	86.7	100.0	83.3	86.7	100.0	100.0
Anchor-Align	30.0	86.7	3.3	100.0	1.7	91.7	10.0	71.7	100.0	98.3	100.0	95.0	86.7	98.3	96.7	96.7	98.3	41.7	100.0	98.3
RIPT-VLA	15.0	58.3	1.7	98.3	0.0	96.7	100.0	60.0	100.0	98.3	96.7	81.7	43.3	76.7	96.7	95.0	98.3	8.3	100.0	100.0
Cosmos Policy	10.0	30.0	30.0	90.0	10.0	96.7	70.0	80.0	93.3	90.0	90.0	76.7	66.7	53.3	96.7	90.0	83.3	6.7	96.7	96.7
FastWAM	12.0	54.0	13.0	90.0	0.0	92.0	48.0	88.0	93.0	94.0	95.0	79.0	59.0	82.0	87.0	82.0	92.0	24.0	88.0	91.0
LingBot-VA	70.0	80.0	100.0	100.0	98.3	98.3	88.3	98.3	100.0	91.7	98.3	100.0	98.3	100.0	100.0	98.3	85.0	1.7	95.0	100.0

Table 10: Static and runtime perturbation results on LIBERO-10. All values are reported in percentages (%).

(a) Static Perturbations

Model	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22
OpenVLA-OFT	30.0	26.7	36.7	0.0	46.7	33.3	91.7	95.0	38.3	80.0	96.7	41.7	95.0	68.3	0.0	6.7	23.3	91.7	90.0	0.0	95.0	85.0
π_0	0.0	1.7	18.3	0.0	31.7	10.0	55.0	43.3	18.3	51.7	50.0	45.0	40.0	41.7	41.7	41.7	31.7	5.0	50.0	1.7	43.3	48.3
$\pi_{0.5}$	36.7	20.0	68.3	76.7	53.3	25.0	71.7	70.0	38.3	83.3	81.7	63.3	75.0	75.0	0.0	50.0	70.0	70.0	71.7	3.3	78.3	80.0
X-VLA	16.7	16.7	13.3	3.3	86.7	50.0	100.0	95.0	28.3	100.0	91.7	55.0	100.0	100.0	48.3	53.3	95.0	25.0	100.0	48.3	75.0	98.3
GR00T N1.7	15.0	23.3	33.3	0.0	61.7	5.0	80.0	80.0	41.7	65.0	71.7	51.7	86.7	80.0	3.3	86.7	50.0	75.0	83.3	0.0	76.7	81.7
MolmoAct2	41.7	28.3	83.3	86.7	60.0	18.3	81.7	78.3	41.7	91.7	95.0	56.7	76.7	76.7	23.3	80.0	60.0	73.3	86.7	0.0	88.3	83.3
Anchor-Align	26.7	20.0	38.3	0.0	36.7	16.7	70.0	68.3	20.0	73.3	63.3	83.3	78.3	73.3	0.0	21.7	31.7	71.7	60.0	3.3	76.7	61.7
RIPT-VLA	30.0	23.3	38.3	0.0	43.3	13.3	91.7	93.3	43.3	83.3	93.3	43.3	91.7	78.3	0.0	15.0	26.7	88.3	91.7	0.0	91.7	86.7
Cosmos Policy	76.7	70.0	33.3	3.3	20.0	33.3	90.0	90.0	66.7	90.0	90.0	100.0	100.0	90.0	46.7	100.0	83.3	93.3	96.7	0.0	96.7	90.0
FastWAM	25.0	6.7	15.0	0.0	78.3	31.7	96.7	95.0	38.3	100.0	100.0	93.3	100.0	100.0	0.0	0.0	86.7	71.7	0.0	0.0	96.7	85.0
LingBot-VA	46.7	51.7	88.3	0.0	83.3	38.3	0.0	98.3	0.0	76.7	80.0	90.0	91.7	85.0	6.7	26.7	98.3	96.7	60.0	91.7	98.3	93.3

(b) Runtime Perturbations

Model	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20
OpenVLA-OFT	0.0	18.3	28.3	96.7	40.0	100.0	45.0	65.0	93.3	86.7	73.3	91.7	98.3	86.7	91.7	93.3	95.0	3.3	46.7	95.0
π_0	0.0	10.0	10.0	41.7	18.3	70.0	41.7	36.7	48.3	38.3	50.0	43.3	30.0	11.7	36.7	41.7	5.0	0.0	15.0	48.3
$\pi_{0.5}$	5.0	36.7	11.7	75.0	43.3	100.0	43.3	20.0	73.3	81.7	66.7	76.7	71.7	68.3	76.7	80.0	68.3	28.3	70.0	83.3
X-VLA	0.0	16.7	31.7	98.3	43.3	100.0	61.7	26.7	98.3	95.0	95.0	98.3	95.0	93.3	98.3	15.0	83.3	60.0	95.0	95.0
GR00T N1.7	0.0	25.0	15.0	81.7	26.7	90.0	25.0	70.0	76.7	75.0	73.3	73.3	83.3	78.3	78.3	85.0	73.3	0.0	50.0	78.3
MolmoAct2	1.7	21.7	18.3	85.0	45.0	100.0	66.7	61.7	78.3	71.7	73.3	75.0	78.3	63.3	81.7	83.3	83.3	25.0	73.3	83.3
Anchor-Align	0.0	8.3	18.3	66.7	21.7	86.7	38.3	38.3	75.0	60.0	58.3	78.3	81.7	65.0	71.7	85.0	66.7	0.0	53.3	68.3
RIPT-VLA	0.0	15.0	25.0	98.3	25.0	100.0	46.7	53.3	91.7	83.3	66.7	80.0	85.0	76.7	80.0	86.7	88.3	0.0	56.7	90.0
Cosmos Policy	40.0	43.3	23.3	96.7	10.0	100.0	96.7	70.0	100.0	93.3	86.7	90.0	90.0	66.7	83.3	90.0	83.3	33.3	83.3	96.7
FastWAM	1.7	5.0	18.3	96.7	45.0	100.0	18.3	90.0	98.3	98.3	93.3	96.7	98.3	83.3	95.0	96.7	95.0	31.7	88.3	96.7
LingBot-VA	21.7	36.7	93.3	95.0	95.0	95.0	71.7	88.3	96.7	85.0	91.7	91.7	96.7	95.0	96.7	96.7	46.7	1.7	100.0	96.7

A.2 Complete Feedback Utilization Results

Table 11 reports all five policies under live and frozen observations, with and without each dynamic perturbation.

Table 11: Complete feedback-utilization results. Success rates are percentages; $0/1$ denotes absence/presence of a perturbation, and live/freeze denotes the visual-observation condition. $G_{\text{feedback}} = (S_{1,\text{live}} - S_{1,\text{freeze}}) - (S_{0,\text{live}} - S_{0,\text{freeze}})$ is measured in percentage points. The two panels cover D01–D10 and D11–D20, respectively.

(a): D01–D10

Metric	**D01**	**D02**	**D03**	**D04**	**D05**	**D06**	**D07**	**D08**	**D09**	**D10**
OpenVLA-OFT										
$S_{0,\text{live}}$	97.1	97.1	97.1	97.1	97.1	97.1	97.1	97.1	97.1	97.1
$S_{0,\text{freeze}}$	45.6	43.8	33.8	50.0	42.5	55.0	48.1	48.1	48.1	48.1
$S_{1,\text{live}}$	5.4	24.3	17.6	95.8	16.8	95.3	38.3	56.7	94.2	87.9
$S_{1,\text{freeze}}$	0.0	3.8	0.0	41.9	13.8	43.6	21.3	41.3	48.1	47.5
$\mathbf{G}_{\text{feedback}}$	**-46.1**	**-32.8**	**-45.8**	**6.9**	**-51.5**	**9.6**	**-31.9**	**-33.6**	**-2.9**	**-8.6**
$\boldsymbol{\pi}_{0.5}$										
$S_{0,\text{live}}$	93.3	93.3	93.3	93.3	93.3	93.3	93.3	93.3	93.3	93.3
$S_{0,\text{freeze}}$	86.9	91.3	80.0	83.8	85.6	92.9	85.6	85.6	85.6	85.6
$S_{1,\text{live}}$	50.4	76.9	19.8	92.9	21.1	97.4	16.3	65.8	92.1	88.8
$S_{1,\text{freeze}}$	41.3	64.4	15.6	85.6	41.3	92.1	38.8	83.8	86.3	86.3
$\mathbf{G}_{\text{feedback}}$	**2.7**	**10.5**	**-9.1**	**-2.3**	**-27.8**	**4.8**	**-30.2**	**-25.6**	**-1.9**	**-5.2**
Anchor-Align										
$S_{0,\text{live}}$	91.3	91.3	91.3	91.3	91.3	91.3	91.3	91.3	91.3	91.3
$S_{0,\text{freeze}}$	78.1	83.8	56.9	72.5	72.5	78.6	74.4	74.4	74.4	74.4
$S_{1,\text{live}}$	16.3	38.2	15.3	88.8	13.9	79.8	12.1	39.2	90.8	75.4
$S_{1,\text{freeze}}$	16.3	30.0	11.3	74.4	25.6	80.0	19.4	57.5	73.1	75.6
$\mathbf{G}_{\text{feedback}}$	**-13.1**	**0.7**	**-30.3**	**-4.4**	**-30.5**	**-12.9**	**-24.2**	**-35.2**	**0.8**	**-17.1**
FastWAM										
$S_{0,\text{live}}$	97.8	97.8	97.8	97.8	97.8	97.8	97.8	97.8	97.8	97.8
$S_{0,\text{freeze}}$	92.5	83.1	52.5	91.9	81.9	94.3	91.9	91.9	91.9	91.9
$S_{1,\text{live}}$	20.7	17.3	17.6	92.1	18.3	94.1	16.6	89.7	94.9	96.0
$S_{1,\text{freeze}}$	23.8	13.1	6.3	91.3	36.3	92.9	30.6	90.6	90.0	91.9
$\mathbf{G}_{\text{feedback}}$	**-8.3**	**-10.4**	**-33.9**	**-5.0**	**-33.9**	**-2.2**	**-19.9**	**-6.8**	**-1.0**	**-1.8**
Cosmos Policy										
$S_{0,\text{live}}$	97.5	97.5	97.5	97.5	97.5	97.5	97.5	97.5	97.5	97.5
$S_{0,\text{freeze}}$	93.1	93.8	74.4	95.6	82.5	94.3	91.9	91.9	91.9	91.9
$S_{1,\text{live}}$	21.7	26.9	21.4	95.8	9.0	97.4	75.8	80.8	97.5	91.7
$S_{1,\text{freeze}}$	15.0	20.0	15.6	94.4	50.0	96.4	85.6	90.0	92.5	90.0
$\mathbf{G}_{\text{feedback}}$	**2.3**	**3.2**	**-17.4**	**-0.4**	**-56.1**	**-2.2**	**-15.4**	**-14.8**	**-0.6**	**-4.0**

Continued on the next page

Table 11 (continued)

(b): D11–D20

Metric	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20
OpenVLA-OFT										
$S_{0,\text{live}}$	97.1	97.1	97.1	97.1	97.1	97.1	97.1	97.1	97.1	97.1
$S_{0,\text{freeze}}$	48.1	48.1	48.1	48.1	41.3	48.1	48.1	41.9	32.5	48.1
$S_{1,\text{live}}$	84.6	79.6	70.1	70.4	91.7	93.8	93.3	18.3	62.9	95.4
$S_{1,\text{freeze}}$	45.0	47.5	46.3	43.8	43.1	41.3	42.5	1.3	43.8	47.5
$\mathbf{G}_{\text{feedback}}$	**-9.4**	**-16.9**	**-25.1**	**-22.3**	**-7.3**	**3.5**	**1.8**	**-38.1**	**-45.4**	**-1.1**
$\boldsymbol{\pi_{0.5}}$										
$S_{0,\text{live}}$	93.3	93.3	93.3	93.3	93.3	93.3	93.3	93.3	93.3	93.3
$S_{0,\text{freeze}}$	85.6	85.6	85.6	85.6	87.5	85.6	88.1	88.1	71.3	86.3
$S_{1,\text{live}}$	86.3	87.5	83.7	83.3	90.4	92.9	87.6	65.3	78.3	94.2
$S_{1,\text{freeze}}$	80.6	84.4	76.9	85.6	87.5	88.1	85.6	19.4	71.3	85.0
$\mathbf{G}_{\text{feedback}}$	**-2.1**	**-4.6**	**-0.9**	**-10.0**	**-2.9**	**-2.9**	**-3.2**	**40.7**	**-15.0**	**2.1**
Anchor-Align										
$S_{0,\text{live}}$	91.3	91.3	91.3	91.3	91.3	91.3	91.3	91.3	91.3	91.3
$S_{0,\text{freeze}}$	74.4	74.4	74.4	74.4	72.5	74.4	88.1	71.3	58.1	74.4
$S_{1,\text{live}}$	73.3	81.3	79.6	79.6	90.8	92.1	85.9	21.9	68.8	88.3
$S_{1,\text{freeze}}$	71.3	75.6	71.3	76.9	83.1	82.5	73.8	12.5	60.6	79.4
$\mathbf{G}_{\text{feedback}}$	**-14.8**	**-11.3**	**-8.5**	**-14.2**	**-11.0**	**-7.3**	**9.0**	**-10.7**	**-25.0**	**-7.9**
FastWAM										
$S_{0,\text{live}}$	97.8	97.8	97.8	97.8	97.8	97.8	97.8	97.8	97.8	97.8
$S_{0,\text{freeze}}$	91.9	91.9	91.9	91.9	80.6	91.9	91.3	83.8	58.8	91.9
$S_{1,\text{live}}$	94.2	88.9	83.8	86.5	93.4	89.7	94.8	38.2	81.2	93.2
$S_{1,\text{freeze}}$	86.3	89.4	77.5	84.4	88.8	90.0	94.4	19.4	61.3	90.6
$\mathbf{G}_{\text{feedback}}$	**2.0**	**-6.3**	**0.4**	**-3.8**	**-12.5**	**-6.2**	**-6.1**	**4.9**	**-19.1**	**-3.3**
Cosmos Policy										
$S_{0,\text{live}}$	97.5	97.5	97.5	97.5	97.5	97.5	97.5	97.5	97.5	97.5
$S_{0,\text{freeze}}$	91.9	91.9	91.9	91.9	83.1	91.9	97.5	84.4	71.3	91.9
$S_{1,\text{live}}$	90.8	87.5	81.7	70.8	92.5	93.3	87.4	32.5	93.3	95.8
$S_{1,\text{freeze}}$	88.8	95.0	81.3	82.5	95.6	92.5	90.6	3.8	76.3	95.6
$\mathbf{G}_{\text{feedback}}$	**-3.5**	**-13.1**	**-5.2**	**-17.3**	**-17.5**	**-4.8**	**-3.3**	**15.7**	**-9.2**	**-5.4**

A.3 Complete Recovery Intervention Results

Figure 7 and Table 12 provide the recovery gains and success rates for Hold, Retry, and Restore (Reset).

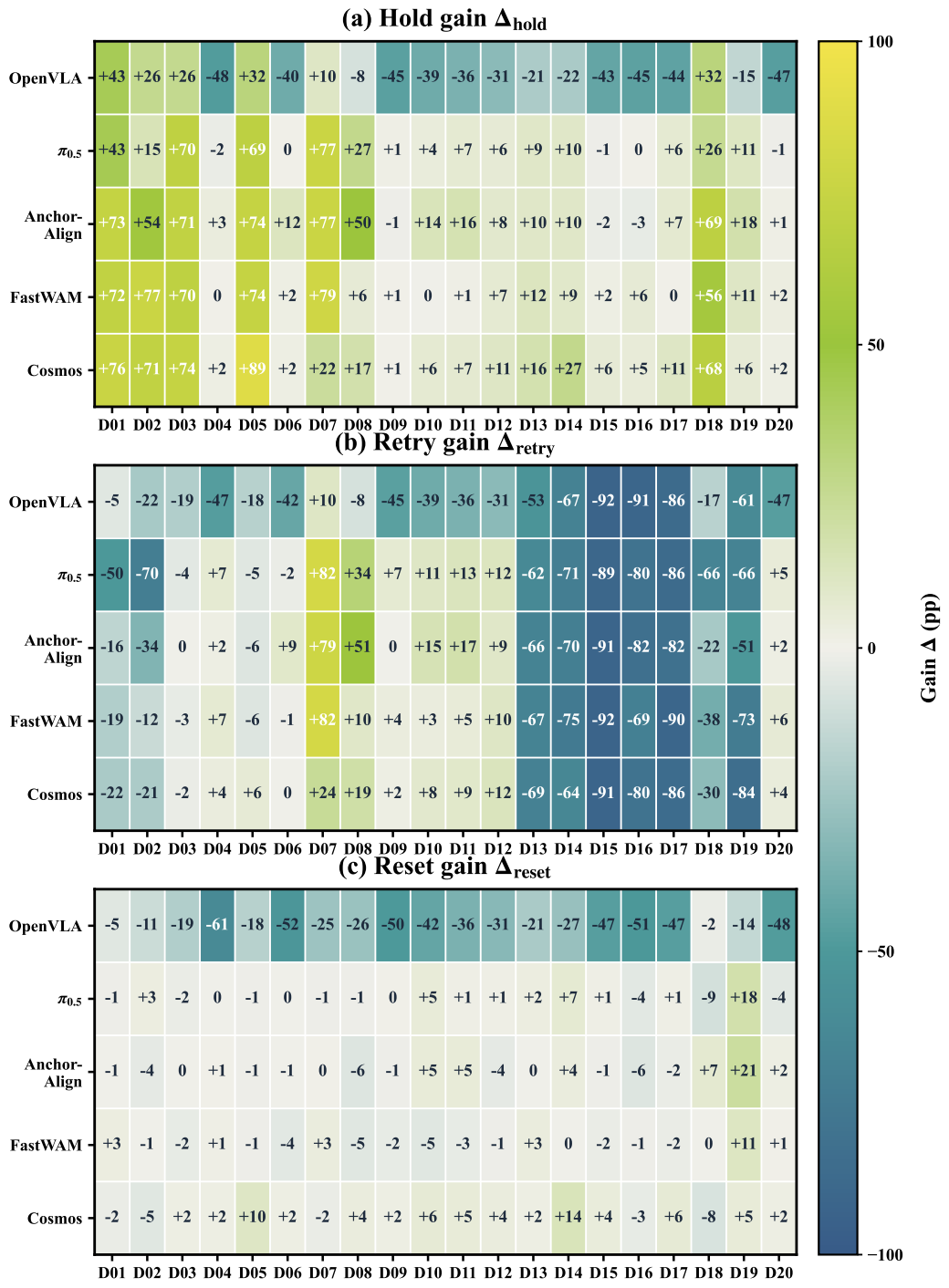

Figure 7: Recovery-intervention gains, $\Delta_I = S_I - S_{\text{continue}}$ (percentage points), for Hold, Retry, and Restore (labeled Reset in the figure). Red denotes improvement over Continue and blue denotes a decrease.

Table 12: Success rates (%) under Hold, Retry, and Restore (Reset) interventions across all dynamic perturbations. Both panels retain the same intervention and model order.

(a) D01–D10

Model	D01	D02	D03	D04	D05	D06	D07	D08	D09	D10
Hold										
OpenVLA-OFT	48.1	48.8	46.3	47.5	49.4	55.7	48.8	48.8	48.8	48.8
$\pi_{0.5}$	93.8	92.5	92.5	90.6	91.3	97.9	93.1	93.1	93.1	93.1
Anchor-Align	89.4	90.0	89.4	91.9	88.8	92.1	89.4	89.4	89.4	89.4
FastWAM	92.5	93.1	90.0	92.5	92.5	96.4	95.6	95.6	95.6	95.6
Cosmos Policy	98.1	98.1	96.3	98.1	98.1	99.3	98.1	98.1	98.1	98.1
Retry										
OpenVLA-OFT	0.0	0.0	0.6	48.8	0.0	53.6	48.8	48.8	48.8	48.8
$\pi_{0.5}$	0.0	7.5	18.1	98.8	16.9	95.7	98.8	99.4	98.8	99.4
Anchor-Align	0.6	1.9	17.5	90.6	9.4	89.3	90.6	90.6	90.6	90.6
FastWAM	1.9	4.4	16.9	98.8	12.5	92.9	98.8	99.4	98.8	99.4
Cosmos Policy	0.0	5.6	20.6	99.4	15.0	97.1	99.4	100.0	99.4	100.0
Reset										
OpenVLA-OFT	0.6	11.3	0.6	34.4	0.0	43.6	13.1	30.6	44.4	45.6
$\pi_{0.5}$	49.4	80.6	20.0	91.9	20.6	97.1	15.0	65.0	92.5	93.8
Anchor-Align	15.6	32.5	17.5	89.4	13.8	79.3	12.5	33.1	90.0	80.6
FastWAM	23.8	15.6	17.5	93.1	17.5	90.0	20.0	84.4	92.5	91.3
Cosmos Policy	20.0	21.9	24.4	97.5	19.4	99.3	73.8	84.4	99.4	97.5

(b) D11–D20

Model	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20
Hold										
OpenVLA-OFT	48.8	48.8	48.8	48.8	48.8	48.8	48.8	48.8	48.1	48.8
$\pi_{0.5}$	93.1	93.1	93.1	93.1	89.4	93.1	94.4	91.9	89.4	93.1
Anchor-Align	89.4	89.4	89.4	89.4	88.8	89.4	92.5	91.3	86.3	89.4
FastWAM	95.6	95.6	95.6	95.6	95.6	95.6	95.0	93.8	91.9	95.6
Cosmos Policy	98.1	98.1	98.1	98.1	98.1	98.1	98.1	98.8	99.4	98.1
Retry										
OpenVLA-OFT	48.8	48.8	16.3	3.1	0.0	3.1	6.9	0.0	1.9	48.8
$\pi_{0.5}$	99.4	99.4	21.9	12.5	1.3	12.5	1.9	0.0	11.9	99.4
Anchor-Align	90.6	90.6	13.1	9.4	0.0	10.0	3.8	0.0	17.5	90.6
FastWAM	99.4	99.4	16.9	11.9	1.3	20.6	4.4	0.0	8.1	99.4
Cosmos Policy	100.0	100.0	13.1	6.9	1.3	13.1	1.9	0.6	9.4	100.0
Reset										
OpenVLA-OFT	48.8	48.1	48.1	43.8	45.0	43.1	46.3	15.0	49.4	47.5
$\pi_{0.5}$	87.5	88.8	86.3	90.0	91.3	88.8	88.8	56.9	96.3	90.0
Anchor-Align	78.8	77.5	79.4	83.1	90.0	86.3	83.1	29.4	90.0	90.0
FastWAM	91.3	87.5	86.9	86.3	91.3	88.8	92.5	38.1	92.5	93.8
Cosmos Policy	95.6	91.3	83.8	84.4	96.9	90.6	93.1	23.1	98.8	97.5

A.4 Complete Robustness Composition Results

Table 13 lists all evaluated pairwise and three-way compositions, grouped by static and dynamic intervention types.

Table 13: Success rates (%) under pairwise and three-way perturbation compositions. Compositions are grouped by perturbation type.

Case	Composition	OpenVLA-OFT	$\pi_{0.5}$	Anchor-Align	FastWAM	Cosmos Policy
			Static + Static			
P01	S11 + S15	30.4	0.0	0.0	5.0	50.8
P02	S09 + S13	12.5	12.5	10.0	9.2	12.9
P03	S19 + S21	82.9	86.3	82.5	5.8	85.4
P04	S08 + S22	82.5	80.8	73.8	76.3	80.4
			Static + Dynamic			
P05	S11 + D01	7.9	56.3	14.6	15.8	21.7
P06	S09 + D13	12.5	12.1	11.3	11.7	11.7
P07	S15 + D18	0.0	0.0	0.0	0.0	0.4
			Dynamic + Dynamic			
P08	D02 + D18	0.0	22.9	0.8	0.0	1.3
P09	D03 + D19	60.0	73.8	67.5	73.8	80.8
P10	D18 + D20	0.0	17.5	0.0	0.0	2.9
			Three-way Compositions			
T01	S11 + S21 + D01	7.1	56.7	14.6	14.6	24.6
T02	S13 + S15 + D18	0.0	0.0	0.0	0.0	0.0
T03	S09 + S22 + D13	12.5	12.9	11.7	10.0	12.1
T04	S13 + D02 + D18	0.0	8.3	0.0	0.0	0.0
T05	S13 + S22 + D09	80.8	86.7	75.0	75.0	86.7
T06	S10 + S18 + D06	83.8	87.1	73.3	80.4	87.1
T07	S11 + S22 + D07	81.3	31.7	12.9	23.8	85.4
T08	S19 + S21 + D14	84.6	85.0	81.3	6.3	85.4
T09	S08 + S13 + D17	69.6	80.0	52.5	76.3	63.8
T10	S14 + S21 + D18	0.0	21.7	0.0	0.0	1.7

B PERTURBATION DESIGN

The following panels illustrate all 22 static and 20 dynamic perturbations. Each row shows one perturbation using aligned views or consecutive frames; the complete definitions are given in Appendix C.

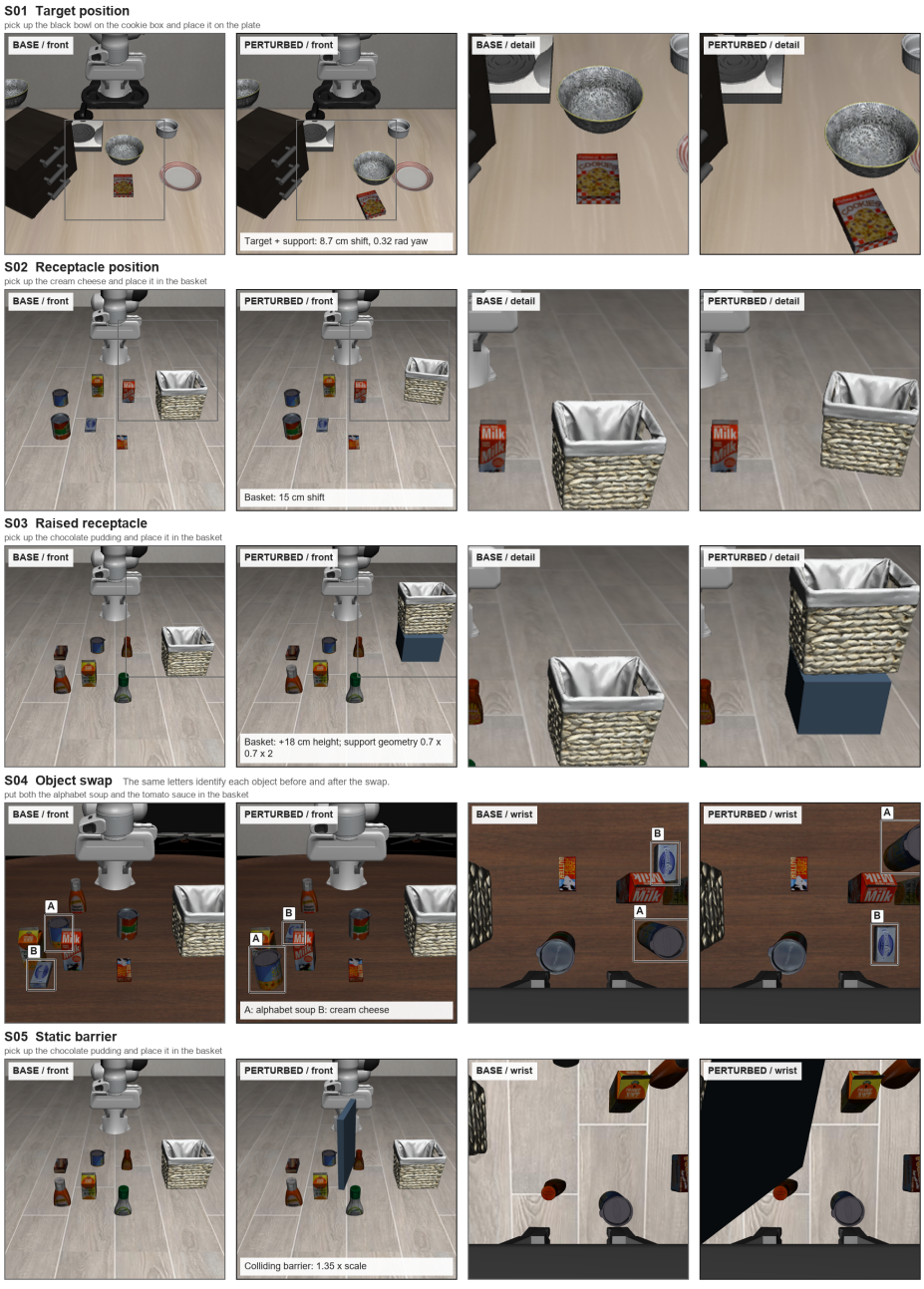

Figure 8: Static perturbations S01–S05. Each row contrasts BASE and the perturbed scene using the external and wrist views.

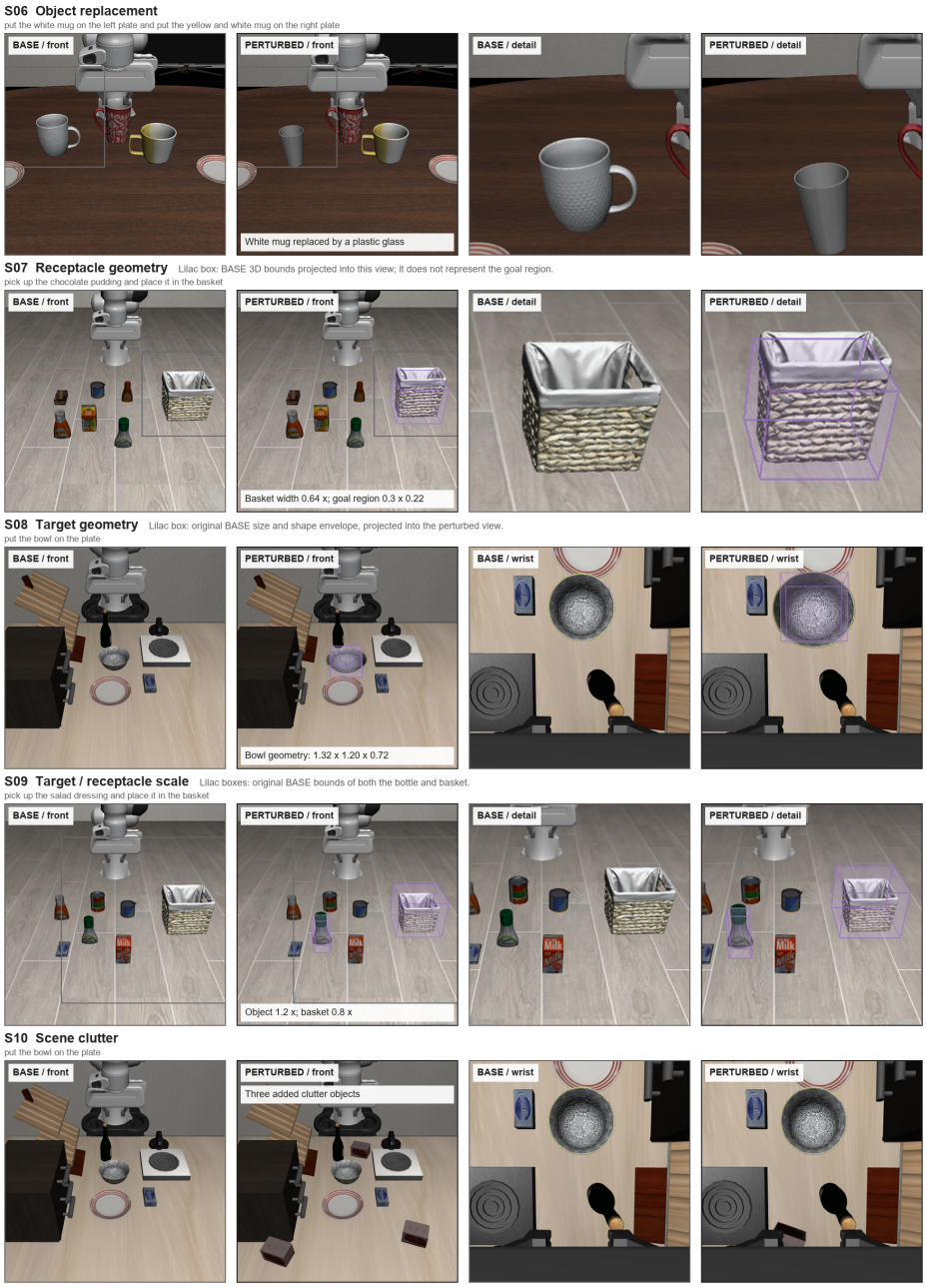

Figure 9: Static perturbations S06–S10.

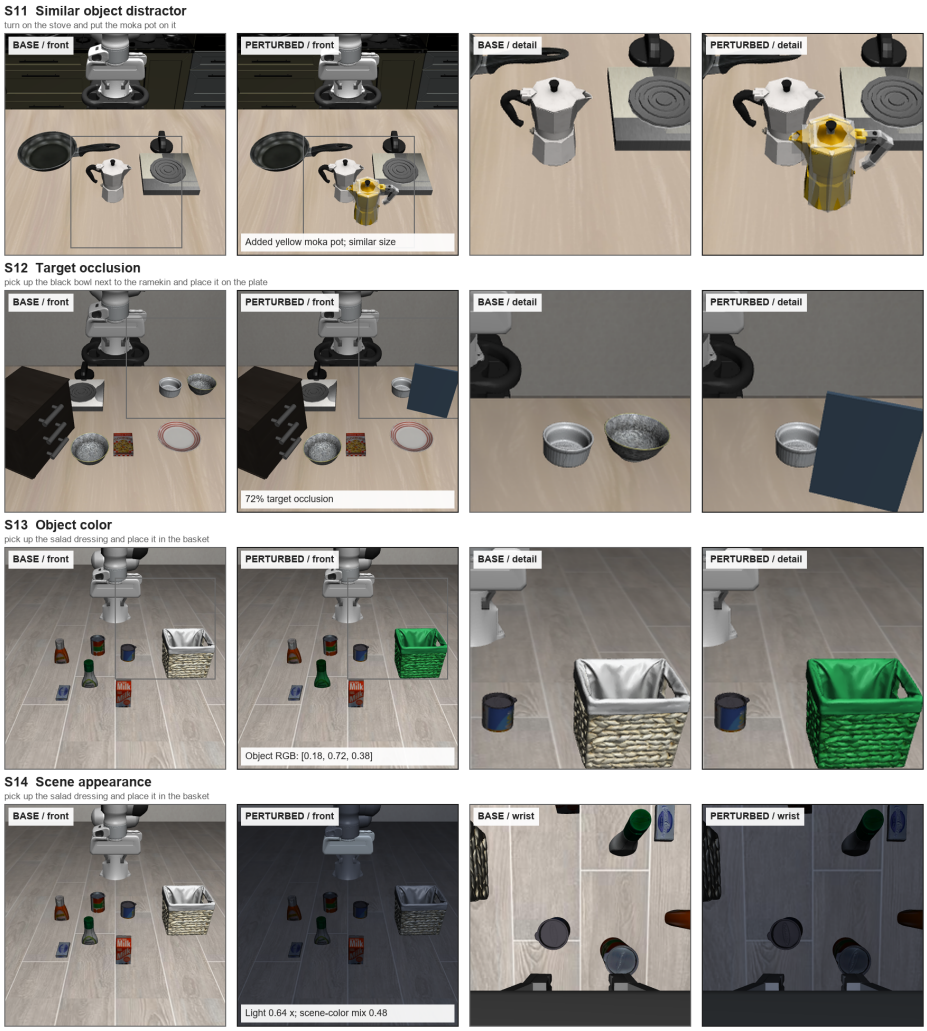

Figure 10: Static perturbations S11–S14.

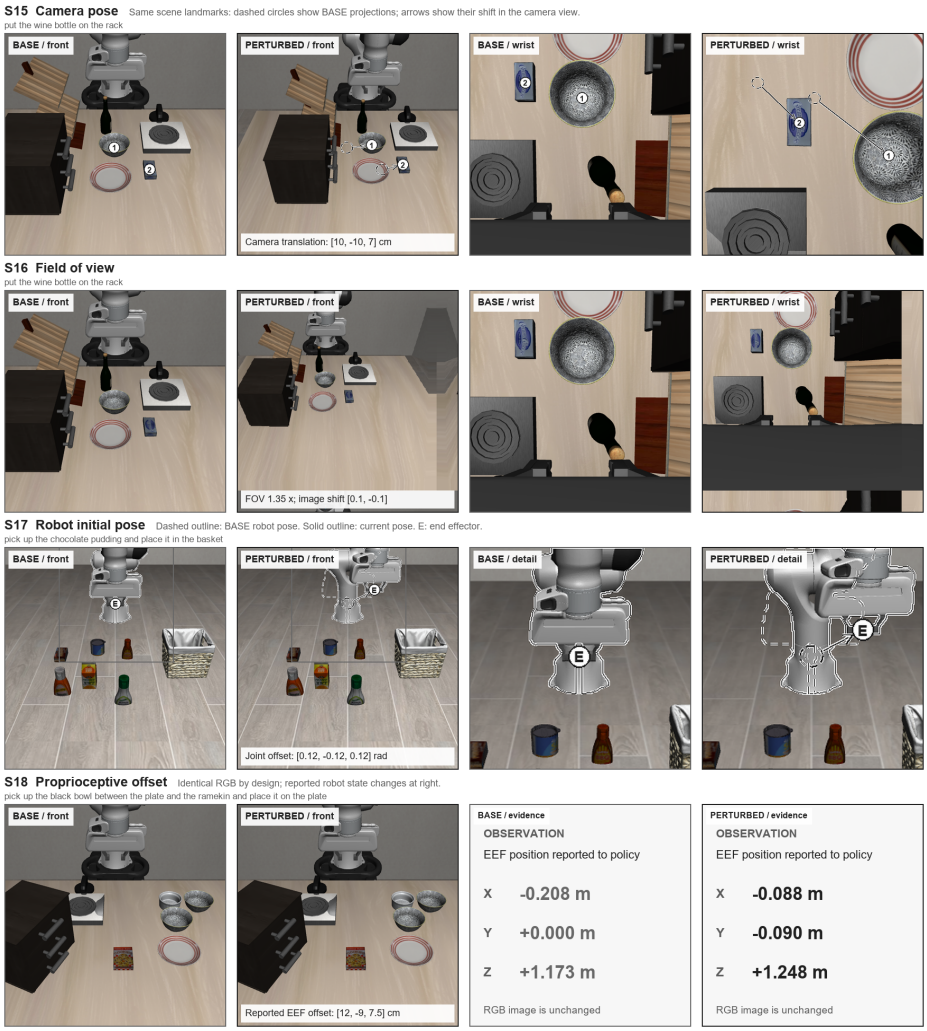

Figure 11: Static perturbations S15–S18.

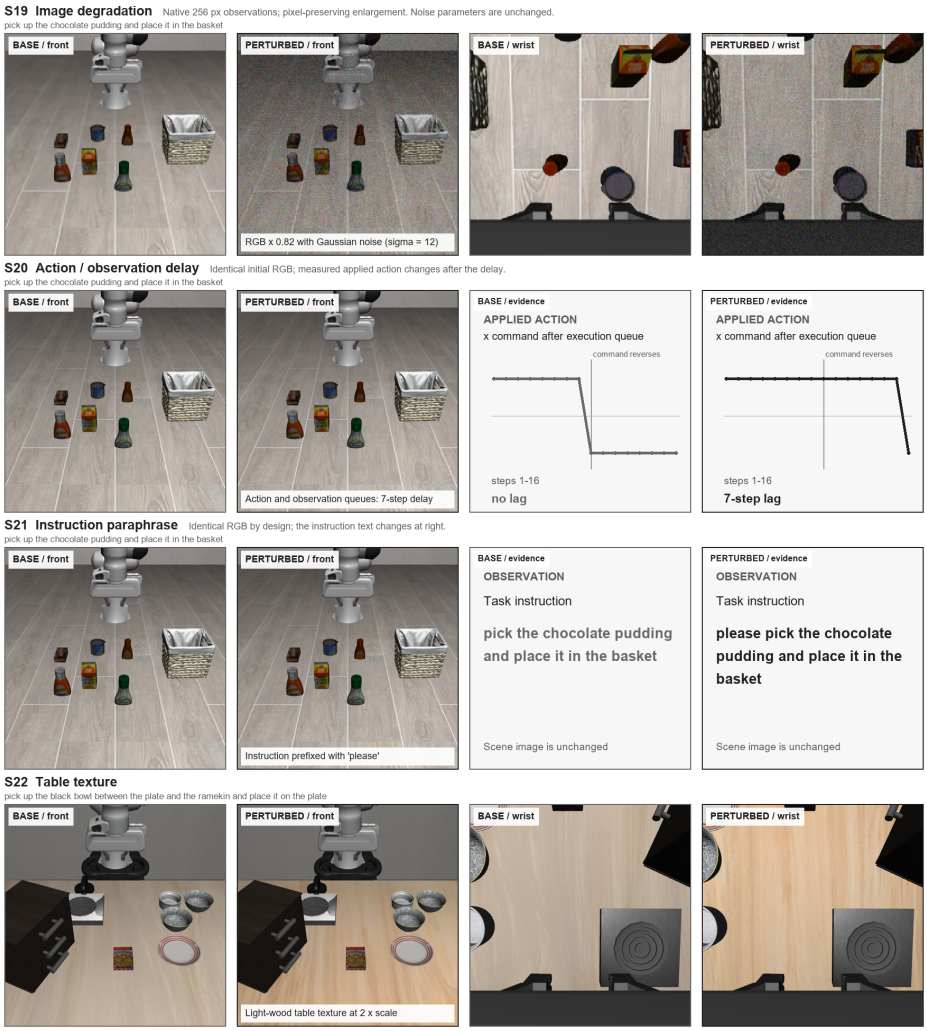

Figure 12: Static perturbations S19–S22.

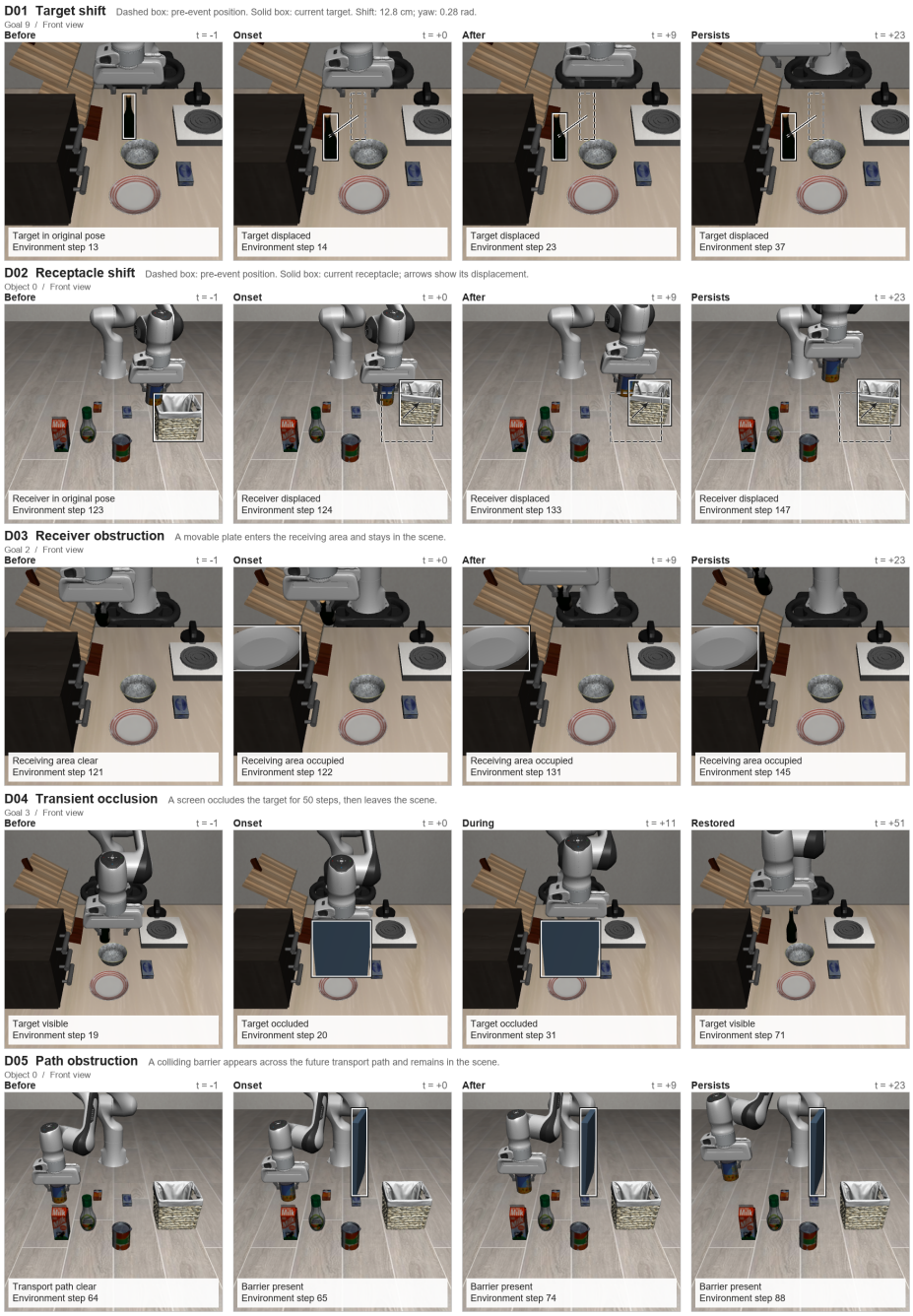

Figure 13: Dynamic perturbations D01–D05. Each row shows the temporal sequence around an intervention, using the views indicated in the panels.



Figure 14: Dynamic perturbations D06–D10.

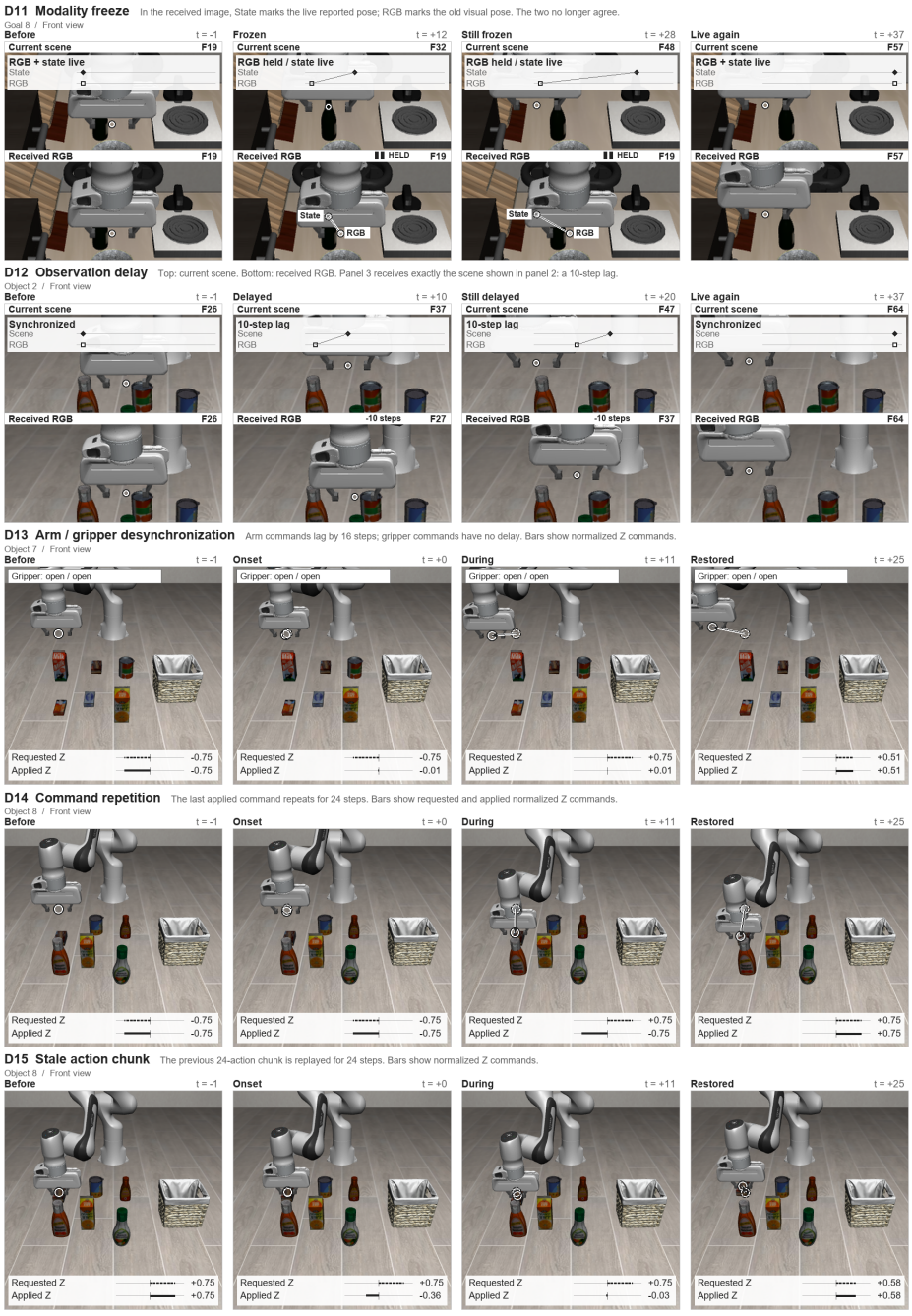

Figure 15: Dynamic perturbations D11–D15.

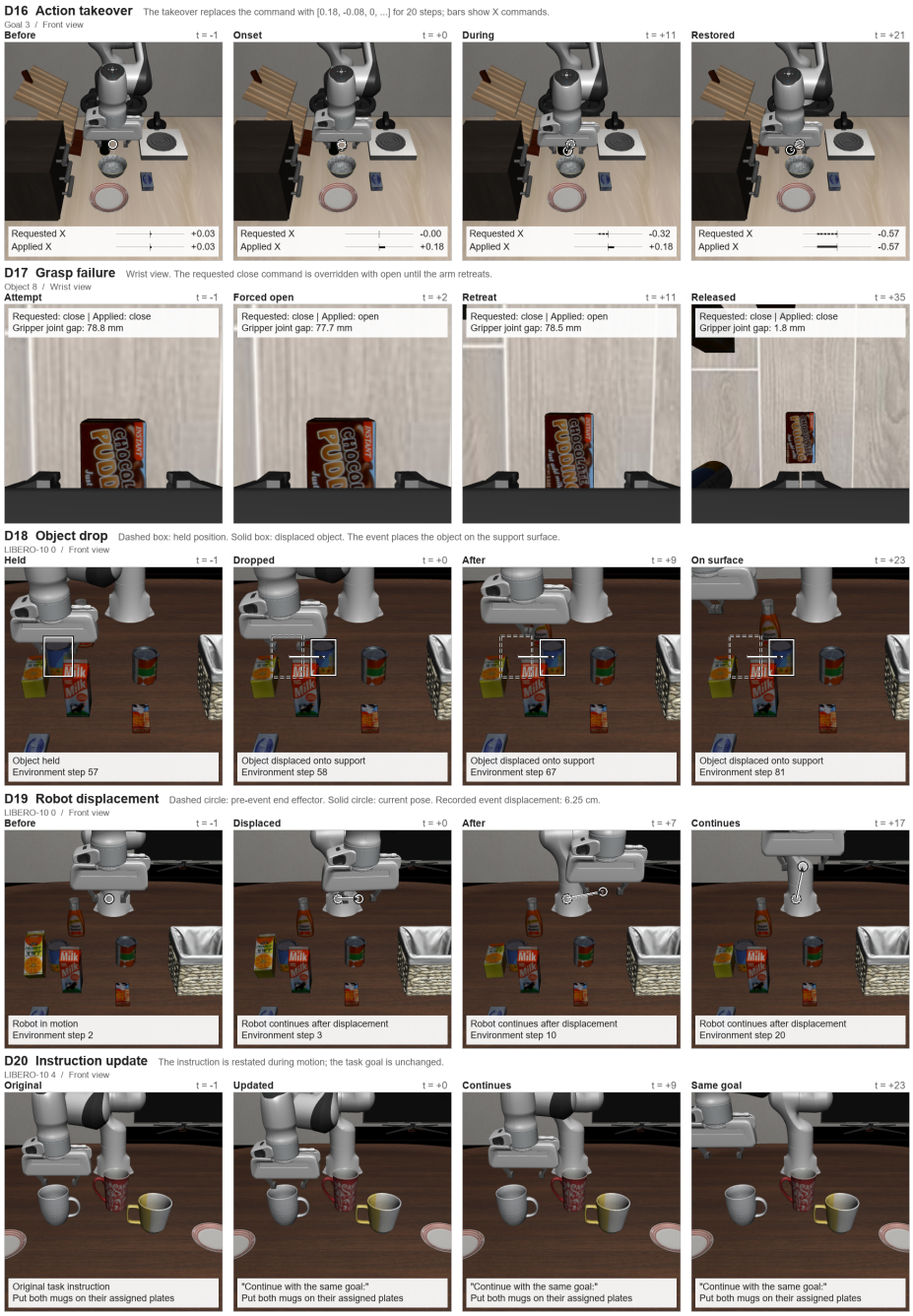

Figure 16: Dynamic perturbations D16–D20.



Figure 17: Representative failure rollouts under perturbations; each row contains two five-frame sequences, read from left to right.

C PERTURBATION CATALOGUE

This section supplements the LIBERO-Pro results with definitions of all 22 static and 20 dynamic perturbations and the task partition used in the controlled eight-task studies. The examples in Appendix B illustrate these definitions; the nominal, unperturbed task is the BASE reference.

Design scope. Perturbations cover object and scene geometry, appearance, sensing, temporal alignment, action execution, robot state, and language. Each type shares a semantic definition across suites; affected objects, magnitudes, directions, and event settings are specified per task. These parameter choices do not define additional perturbation types.

Task validity. The original goal and success predicate are preserved. Geometry and state changes retain reachable, collision-free configurations and feasible task completion. Planar pose changes combine horizontal translation and yaw; pitch and roll are excluded from this category. Physical parameters such as mass, inertia, friction, stiffness, and damping are not varied. Grasp loss and robot displacement are represented by injected outcome states or bounded control interventions, rather than simulated impact or falling forces.

Initialization and runtime events. Static perturbations are introduced at reset; altered scene or fault settings persist while normal robot and object motion remains possible. Dynamic perturbations are triggered at task-relevant phases, such as approach, grasp, transport, or placement. Trigger conditions, duration, and release rules are specified by each case. Restoring a sensor or control channel does not imply rolling back the complete scene. Experiment-specific freeze or waiting durations are distinct from these general mechanism definitions.

C.1 CATEGORY DEFINITIONS AND ASSIGNMENT PRINCIPLE

We assign each perturbation to one category according to its primary intervention target. The categories organize perturbations by their dominant operational effect and are not mutually exclusive descriptions of every downstream consequence. *Condition* perturbations modify task-relevant object or receiver configurations, relative layouts, or geometric constraints on approach and placement. *Environment* perturbations modify object instances or scene context, including clutter, distractors, occlusion, visual-domain appearance, support-surface appearance, and external visual conditions such as camera motion or lighting changes. *Observation* perturbations modify the formation, quality, availability, or synchronization of the inputs delivered to the policy. *Execution* perturbations modify control timing or action execution. *Robot* perturbations modify robot initialization, calibration, or robot-related state and manipulation outcomes. *Language* perturbations modify instruction wording while preserving task semantics. Some perturbations may affect multiple aspects of task execution; each is assigned according to its primary intervention target. This mapping is used consistently to compute the category averages in Tables 1–3.

Table 14: Mapping of static and runtime perturbations to evaluation categories.

Category	Static perturbations	Runtime perturbations
Condition	S01–S05, S07–S09	D01–D06
Environment	S06, S10–S14, S22	D07–D09
Observation	S15–S16, S19	D10–D13
Execution	S20	D14–D16
Robot	S17–S18	D17–D19
Language	S21	D20

C.2 STATIC PERTURBATIONS

Table 15 summarizes each initialization-time intervention and the constraints that preserve the manipulation task.

Table 15: Static perturbation definitions. Interventions are introduced at reset. Modified geometry or fault settings persist; initial-state changes do not prevent normal subsequent motion. Alternatives within a row are selected by the task configuration.

ID	Perturbation	Intervention	Constraint
S01	Target planar pose	Translate the target in the horizontal plane and rotate it in yaw.	Preserve the target identity and intended goal.
S02	Receiver planar pose	Translate or rotate the receiver to change the placement destination and approach.	Preserve receiver identity and task goal.
S03	Receiver or support height	Change receiver or support elevation using a support platform or a coupled layout adjustment.	Retain a collision-free initial layout and feasible placement.
S04	Object position swap	Swap the planar positions of two objects; use a safe layout offset when a direct swap is unsuitable.	Keep object identities, task roles, and goal unchanged.
S05	Restricted approach or path	Add geometry that constrains approach or transport.	Retain valid inverse kinematics and a complete collision-free route.
S06	Instance replacement	Replace the target with a same-class or unseen asset specified by the task configuration.	Preserve semantic class and graspability.
S07	Functional-part geometry	Change an opening, rim, or receiving region used for grasping or placement.	Retain the intended functional role.
S08	Size or aspect ratio	Apply uniform or anisotropic target scaling.	Change geometric extent while preserving target identity.
S09	Opening clearance and pose	Jointly change object/container geometry and the relative insertion configuration.	Use collision geometry with full containment, positive clearance, and sufficient depth.
S10	Irrelevant clutter	Add irrelevant objects or change their spatial density.	Preserve target and receiver; support clutter without interpenetration.
S11	Similar distractors	Add objects of the same class or similar color or shape.	The intended target remains the same.
S12	Persistent occlusion	Place and retain a foreground occluder between the camera and the target region.	Keep the occlusion intervention active during the rollout.
S13	Object appearance	Change target or receiver color, texture, reflectance, or visual surface state.	Preserve task semantics; do not vary physical material parameters.
S14	Contrast or scene domain	Change object-background contrast or the visual scene domain.	Preserve the intended manipulation task.
S15	Camera extrinsics	Apply fixed camera position/orientation offsets or cross-camera calibration offsets.	Offsets are fixed for the episode; affected cameras are case-specific.
S16	Camera image formation	Change field of view, resolution, cropping, or lens distortion.	The configured imaging transformation remains active.

Continued on the next page

Table 15 (continued)

ID	Perturbation	Intervention	Constraint
S17	Initial robot state	Change initial joints, end-effector pose/orientation, or gripper opening.	Modify initialization; the robot then executes normally.
S18	Robot calibration	Offset the base, tool center point, joint zero, or grasp center.	Keep the configured calibration error fixed.
S19	Observation quality	Apply compression artifacts, noise, blur, exposure contamination, or missing depth.	Affect only the configured observation channels.
S20	Fixed timing or action window	Apply a fixed delay or change control rate, replanning rate, or the action execution window.	Use the configured timing intervention; distinguish it from model defaults.
S21	Instruction paraphrase	Replace the instruction with a semantic paraphrase.	Preserve goal, referenced objects, and action order.
S22	Support texture	Replace the visual texture of the work surface.	Preserve collision geometry, friction, and initial object poses.

C.3 Dynamic Perturbations

Table 16 summarizes runtime interventions, including their trigger phases and event lifetimes.

Table 16: Dynamic perturbation definitions. Trigger phase and event lifetime are part of each case configuration. A restored sensor or control channel does not imply rollback of the complete scene. IDs match the paper visualizations.

ID	Perturbation	Intervention	Trigger and lifetime
D01	Target pose update	Change target horizontal position or yaw, requiring updated localization and approach.	Approach/pre-grasp; retain the update as configured.
D02	Receiver pose update	Move or rotate the receiver without changing the task goal.	Post-grasp/pre-place; retain the update as configured.
D03	Receiver occupation	Introduce an occupying or blocking object near the receiving region.	Pre-place; occupation/release follows the case configuration.
D04	Temporary occlusion	Insert an occluder between camera and target, then remove it.	Task phase; enter, hold for a window, then exit.
D05	Path obstacle	Place an obstacle on the upcoming transport route while preserving an alternative collision-free path.	Transport; persistence follows the case configuration.
D06	Distractor rearrangement	Move irrelevant objects or rearrange local clutter; exclude the target and receiver.	Task phase; apply the configured scene update.
D07	Camera motion	Change camera position to represent shake, drift, or a persistent offset.	Task phase; transient or persistent, as configured.
D08	Camera dropout	Invalidate a selected camera observation stream.	Task phase; restore the stream after the event.
D09	Lighting or exposure	Change RGB illumination, shadow, or exposure.	Task phase; restore after the event window.

Continued on the next page

Table 16 (continued)

ID	Perturbation	Intervention	Trigger and lifetime
D10	Frame-stream fault	Drop, repeat, or freeze configured RGB/depth frames.	Task phase; apply the fault during the configured window.
D11	Modality misalignment	Offset timestamps or freeze a modality while other modalities may continue to update.	Task phase; configured modality and event window.
D12	Latency spike	Temporarily delay observation delivery or inference results.	Task phase; restore normal timing after the spike.
D13	Arm–gripper desynchronization	Apply different delays or timing variations to arm and gripper commands.	Task phase; configured timing window.
D14	Command loss or repetition	Disrupt the action stream; the listed implementation repeats the previous action.	Task phase; short command-fault window.
D15	Stale action chunk	Replay an old action chunk or interrupt replanning, temporarily executing outdated actions.	State change; configurable chunk length and stale duration.
D16	Bounded takeover	Inject contact-free bounded actions or a reachable state offset, respecting joint/workspace limits.	Task phase; temporary intervention, then return control.
D17	Grasp failure result	Hold the gripper open on the first grasp attempt while retaining the possibility of retrying.	Grasp attempt; release after about 0.07 m of retreat or window expiry.
D18	Reachable drop state	Place the transported target at a reachable, collision-free recovery location; do not simulate falling forces.	Transport; inject a recoverable outcome state.
D19	Robot joint displacement	Inject one reachable joint displacement without self-collision and within joint/workspace limits.	Robot in motion; one state displacement, not a force simulation.
D20	Instruction restatement	Restate or clarify the instruction while preserving goal, objects, and required action order.	Task phase; apply the equivalent instruction update.

C.4 Training and Evaluation Task Partition

For the controlled eight-task studies, two BASE tasks per suite are reserved for evaluation; the remaining eight per suite form the training pool (Table 17). The exact evaluation instructions are listed in Table 18. These roles do not imply that the evaluated pretrained checkpoints have never encountered the evaluation tasks. Demonstration selection and trajectory counts are specified separately for each training experiment. Long task 8 with D06 is not applicable and is excluded from the corresponding dynamic-evaluation denominator.

Table 17: Task partition for the controlled eight-task studies. IDs are zero-based indices in the pinned release manifest. The training pool is the complement of the evaluation tasks within each suite; it does not specify which demonstrations were actually used in a training run.

Suite	Evaluation IDs	Training-pool IDs
Spatial	0, 8	1, 2, 3, 4, 5, 6, 7, 9
Object	1, 8	0, 2, 3, 4, 5, 6, 7, 9
Goal	3, 6	0, 1, 2, 4, 5, 7, 8, 9
Long	5, 8	0, 1, 2, 3, 4, 6, 7, 9

Table 18: The eight evaluation tasks. Instructions are taken directly from the BASE scene descriptions, with whitespace normalized.

Suite	ID	BASE task instruction
Spatial	0	Pick the akita black bowl between the plate and the ramekin and place it on the plate
Spatial	8	Pick the akita black bowl next to the plate and place it on the plate
Object	1	Pick the cream cheese and place it in the basket
Object	8	Pick the chocolate pudding and place it in the basket
Goal	3	Open the top layer of the drawer and put the bowl inside
Goal	6	Put the cream cheese on the bowl
Long	5	pick up the book and place it in the back compartment of the caddy
Long	8	put both moka pots on the stove

Table 19: The 32 training tasks. Instructions are taken directly from the BASE scene descriptions, with whitespace normalized.

Suite	ID	BASE task instruction
Spatial	1	Pick the akita black bowl next to the ramekin and place it on the plate
Spatial	2	Pick the akita black bowl from table center and place it on the plate
Spatial	3	Pick the akita black bowl on the cookies box and place it on the plate
Spatial	4	Pick the akita black bowl in the top layer of the wooden cabinet and place it on the plate
Spatial	5	Pick the akita black bowl on the ramekin and place it on the plate
Spatial	6	Pick the akita black bowl next to the cookies box and place it on the plate
Spatial	7	Pick the akita black bowl on the stove and place it on the plate
Spatial	9	Pick the akita black bowl on the wooden cabinet and place it on the plate
Object	0	Pick the alphabet soup and place it in the basket
Object	2	Pick the salad dressing and place it in the basket
Object	3	Pick the bbq sauce and place it in the basket
Object	4	Pick the ketchup and place it in the basket
Object	5	Pick the tomato sauce and place it in the basket
Object	6	Pick the butter and place it in the basket
Object	7	Pick the milk and place it in the basket
Object	9	Pick the orange juice and place it in the basket

Continued on the next page

Table 19 (continued)

Suite	ID	BASE task instruction
Goal	0	Open the middle layer of the drawer
Goal	1	Put the bowl on the stove
Goal	2	Put the wine bottle on the top of the drawer
Goal	4	Put the bowl on the top of the drawer
Goal	5	Push the plate to the front of the stove
Goal	7	Turn on the stove
Goal	8	Put the bowl on the plate
Goal	9	Put the wine bottle on the rack
Long	0	put both the alphabet soup and the tomato sauce in the basket
Long	1	put both the cream cheese box and the butter in the basket
Long	2	turn on the stove and put the moka pot on it
Long	3	put the black bowl in the bottom drawer of the cabinet and close it
Long	4	put the white mug on the left plate and put the yellow and white mug on the right plate
Long	6	put the white mug on the plate and put the chocolate pudding to the right of the plate
Long	7	put both the alphabet soup and the cream cheese box in the basket
Long	9	put the yellow and white mug in the microwave and close it