DO LANGUAGE AND VISUAL PERTURBATIONS COMPOUND? INTERACTION EFFECTS AND BACKEND SENSITIVITY IN COMPACT VISION-LANGUAGE-ACTION MODEL EVALUATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Robustness evaluations of vision-language-action (VLA) models typically report per-perturbation success rates from a modest number of episodes, treating each episode's outcome as fixed and each perturbation as separable. Both assumptions bear directly on the validity of the resulting benchmark numbers. We test them in MiniVLA, a compact VLA with roughly 1B parameters, on LIBERO-90, using a pre-specified 3×3 factorial design that crosses instruction and visual perturbations over 7 tasks (630 episodes). Individual episode outcomes are fragile: across 70 pairs of episodes that share the task, initial state, and instruction and run greedily on the same machine and backend, shifting only the random seed changes the outcome in 26% of pairs and the episode length in 63%, while aggregate success moves by two episodes. A preliminary comparison suggests that switching the inference backend at fp32 has a similarly sized effect. Within the factorial, visual perturbation dominates. Mild lighting and camera changes reduce success from 57% to 20%, while paraphrased instructions have little effect. Mismatched instructions drive success on the original task to zero, so this model does not ignore language, unlike recent reports for other VLAs. For paraphrase combined with mild visual perturbation, the only pairing with room above zero success, we find no detectable compounding, and we show how floor effects cap the super-additive loss such a design can detect. Sampled rollouts are no more diverse than a greedy policy perturbed with small Gaussian action noise. We recommend that small-sample VLA robustness evaluations report seeds and the inference backend, treat outcomes at a fixed initial state as stochastic by repeating episodes and reporting interval estimates, and test perturbations jointly at strengths that leave headroom above zero success.

1 INTRODUCTION

Vision-language-action (VLA) models map camera observations and a natural-language instruction directly to robot actions (Brohan et al., 2023; Kim et al., 2024; Belkhale & Sadigh, 2024). Their robustness is commonly assessed in simulation, frequently on LIBERO (Liu et al., 2023) and its extensions, by perturbing a factor of an episode, such as the camera pose, the lighting, or the wording of the instruction, and reporting the success rate under each perturbation, usually one factor at a time (Fei et al., 2026; Zhou et al., 2025; Guo et al., 2026; Zhang et al., 2026; Kim et al., 2026). Each rate typically comes from tens of episodes per task, for example 50 per task in LIBERO-PRO (Zhou et al., 2025). Kress-Gazit et al. (2024) make a related observation about robot learning in general, where success rates are often reported with little or no information about the number of trials, the initial conditions, or the success criteria.

Reading these numbers at face value relies on two assumptions that are seldom tested. The first, which we call *additivity*, is that the effect of each perturbation can be measured in isolation, so that robustness to two co-occurring shifts follows from robustness to each. The second, *reproducibility*,

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.

is that an episode is determined by its task, initial state, and instruction, so that one greedy run per initial state measures the policy there (Figure 1 tests this) and results obtained on different machines can be compared. If perturbations compound, per-axis success rates overstate robustness whenever shifts co-occur. If outcomes depend on incidental details such as the seed or the execution environment, a small difference between two conditions, or between two papers, may reflect those details rather than what the model does.

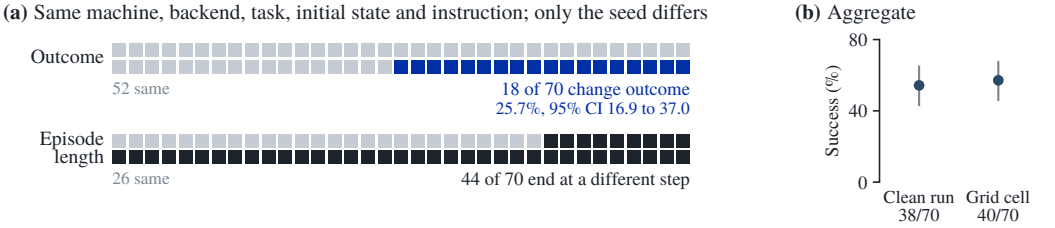

Figure 1: **Changing only the seed changes a quarter of episode outcomes but leaves aggregate success nearly unchanged.** (a) 70 pairs of clean episodes with the same task, initial state, and instruction, run greedily with MPS in fp32 on the same machine. The two episodes in a pair differ only in the seed. Each square is one pair. 18 pairs change outcome (top strip, highlighted) and 44 end at a different step (bottom strip, highlighted). The two strips are separate counts. (b) Success of the two sets of 70 episodes, with Wilson 95% intervals.

Compact VLAs put the second assumption under particular strain, because they are built to run cheaply. MiniVLA is about seven times smaller than OpenVLA and runs inference 2.5 times faster (Belkhale & Sadigh, 2024), and SmolVLA is designed to train on a single GPU and deploy on consumer-grade GPUs or even CPUs (Shukor et al., 2025). This makes it practical to evaluate the same model on CUDA, CPU, or Apple MPS backends, but results from different backends can then be compared as if they measured the same thing. The most closely related evidence comes from LLM inference, where greedy decoding produces different outputs across GPU types, GPU counts, and batch sizes, and the resulting accuracy differences are large in bf16 and nearly vanish in fp32 (Yuan et al., 2025). To our knowledge, no comparable study exists for VLA policies in closed-loop control, in which each action changes the next observation.

For the first assumption, additivity, LIBERO-Plus (Fei et al., 2025) finds that pairs of visual and physical perturbations degrade OpenVLA-OFT more than their individual effects predict, but it excludes language from this analysis. The same work, together with LIBERO-PRO (Zhou et al., 2025), reports that the models they evaluate often ignore the instruction.

In this paper, we test both assumptions on MiniVLA (Belkhale & Sadigh, 2024), a VLA with roughly 1B parameters, on LIBERO-90. All episodes and the analysis plan were fixed before data collection. We frame the work as a study of evaluation methodology: our recommendations rest on the reproducibility and floor-effect results and do not depend on whether perturbations compound. We make the following contributions:

- We show that individual episode outcomes are fragile. On the same machine and backend, changing only the seed changes 18 of 70 outcomes while aggregate success moves by two episodes, and a preliminary comparison suggests that switching the inference backend at fp32 has a similar effect (Figure 1, Section 5).
- We run a pre-specified 3×3 factorial experiment that crosses instruction and visual perturbations (630 episodes over 7 tasks) and analyze it with a Firth-penalized mixed-effects logistic regression. We find no detectable interaction between paraphrase and mild visual perturbation, and we show how floor effects bound the interaction size such a design can detect (Section 4).
- We show that the model conditions on its instruction. Replacing the instruction with that of another task reduces success on the original task to 0 of 210 episodes, in contrast to recent reports for other VLAs (Section 4).

- We compare sampling with greedy decoding plus Gaussian action noise and find that sampling adds about as much per-step action variability as one to three bin widths of noise (Section 6).
- Based on these results, we give recommendations for small-sample VLA robustness evaluation (Table 1, Section 7).

2 Related Work

Robustness benchmarks for VLAs. LIBERO (Liu et al., 2023) is a widely used simulated benchmark for VLAs, and several recent benchmarks build on it or add controlled perturbations. LIBERO-Plus (Fei et al., 2026) varies seven factors, including camera viewpoint, lighting, and instruction wording, and evaluates ten models. LIBERO-PRO (Zhou et al., 2025) perturbs objects, initial positions, instructions, and environments, and finds that models trained on LIBERO often replay memorized trajectories. RobustVLA (Guo et al., 2026) evaluates 17 perturbations across four modalities, including mixed perturbations, and proposes a training method to improve robustness to them. VLA-Arena (Zhang et al., 2026) defines graded language and visual perturbations applicable to any task. These benchmarks report success rates per perturbation or per combination. To our knowledge, LIBERO-Plus is the only one of these that statistically tests for interactions. In its extended version (Fei et al., 2025), it computes the covariance between perturbation indicators among successful episodes, tests it with a chi-squared test, and reports consistent negative interactions among visual and physical factors for OpenVLA-OFT. Language is excluded from this analysis. We test the pairing of language and vision with a pre-specified model and report an interval on the size of the interaction.

Do VLAs use language? Several studies question whether VLAs condition on the instruction at all. Fei et al. (2025) report that removing the instruction barely changes OpenVLA-OFT's success on the LIBERO-Object suite, and that models tend to execute the original task when the instruction's goal object is replaced. Zhou et al. (2025) show a LIBERO-trained model producing nearly the same trajectory when its instruction is replaced by meaningless tokens, and Fang et al. (2026) attribute such counterfactual failures to visual shortcuts induced by dataset biases. At the same time, instruction wording can matter a great deal. LIBERO-Para (Kim et al., 2026) reports drops of 22 to 52 percentage points under paraphrasing across seven VLA configurations. Our mismatch condition tests a basic version of the question, namely whether the instruction of another task prevents the model from completing the original one.

Reliable evaluation. In deep reinforcement learning, reported results depend heavily on random seeds and implementation details, not only on the algorithm (Henderson et al., 2018; Engstrom et al., 2020), and Agarwal et al. (2021) show that point estimates from a few runs can be misleading without interval estimates. Large-scale studies of GANs (Lucic et al., 2018) and prompt formatting in LLMs (Sclar et al., 2024) show that choices outside the method, such as the hyperparameter search budget or prompt format, can change results by amounts comparable to the differences usually attributed to the method itself. For robot learning, Kress-Gazit et al. (2024) recommend reporting the experimental conditions and a statistical analysis alongside success rates, and Snyder et al. (2025) propose sequential tests for comparing policies with small numbers of trials.

Numerical nondeterminism. Floating-point addition is not associative, so changing the order of a reduction can change the last bits of the result. During training, even a one-bit change in the initial parameters can lead to substantially different models (Summers & Dinneen, 2021). At inference time, Yuan et al. (2025) find that greedy LLM decoding diverges across GPU types, GPU counts, and batch sizes, mostly in bf16, and He & Thinking Machines Lab (2025) trace the nondeterminism users observe in LLM servers to kernels whose results depend on batch size. In text generation, a single changed token changes the context for every later token, so the rest of the output can differ. Closed-loop control adds a physical loop: a changed action affects the next observation only through the simulated dynamics, which can damp small differences or amplify them through contact.

3 EXPERIMENTAL SETUP

3.1 MODEL AND BENCHMARK

We evaluate MiniVLA (Belkhale & Sadigh, 2024), which replaces the 7B language backbone of OpenVLA (Kim et al., 2024) with Qwen2.5-0.5B while keeping OpenVLA's fused DINOv2 and SigLIP vision encoder (Karamcheti et al., 2024), for about 1B parameters in total. We use the public LIBERO-90 checkpoint `minivla-libero90-prismatic` at training step 122,500. The model receives a single third-person image and the instruction, with no image history, wrist camera, or action chunking, and predicts seven action tokens per step, one per action dimension, each of which selects one of 256 bins. Preprocessing follows the LIBERO evaluation code released with the model, which applies a center crop at scale 0.9 and normalizes and inverts the gripper action.

LIBERO-90 (Liu et al., 2023) contains 90 tabletop manipulation tasks, each with a fixed set of initial states. We use robosuite 1.4.0 (Zhu et al., 2020) with MuJoCo (Todorov et al., 2012) pinned to version 2.3.2. Before this pin, a seeded episode rerun on another machine with MuJoCo 3.8.0 reached the same outcome but followed a different trajectory. Each episode begins with 10 settle steps in which the robot holds still and ends upon success or after at most 400 policy steps.

Unless stated otherwise, we decode greedily and run inference in fp32 with PyTorch's MPS backend on Apple silicon. We use CPU inference only for the preliminary backend comparison in Section 5, and sampling only in Section 6. Experiments ran on an M5 Pro with 24 GB of memory (macOS 26.5.2, PyTorch 2.14.0) and an Apple M3 (macOS 26.6.2, PyTorch 2.11.0). An M4 Pro with 48 GB of memory (macOS 26.6.2, PyTorch 2.14.0) was used only for the preliminary backend comparison. All machines used transformers 4.40.1, NumPy 1.26.4, robosuite 1.4.0, and MuJoCo 2.3.2. None of our experiments use CUDA (Section 8).

3.2 CLEAN BASELINE

Without perturbations, the model succeeds in 39 of 80 episodes on the M5 Pro (8 tasks with 10 initial states each; Table 4). This is lower than the 61.4% reported by Belkhale & Sadigh (2024), but the two numbers are not comparable. Theirs averages over all 90 tasks, whereas we selected tasks to span high, medium, and low success. Our runs also use Apple MPS in fp32 and one seed per episode, and Section 5 shows that the seed alone changes about a quarter of individual outcomes.

3.3 PERTURBATIONS

Language. Each episode receives one of three instructions: the original LIBERO instruction, a meaning-preserving paraphrase, or a mismatch. A paraphrase keeps the original object and destination and changes the verb and phrasing. A mismatch is an instruction from another LIBERO-90 task, chosen verbatim to share no object, verb, or destination with the original. Appendix A lists all instructions.

Vision. Each episode receives one of three visual levels (none, mild, or strong). The mild and strong levels scale the scene's diffuse and ambient lighting by 0.5 and 0.25, respectively, and offset the third-person camera position with Gaussian noise with a standard deviation of 3 cm and 8 cm per coordinate. We draw the camera offset once per episode from the episode's seed. Both are applied after the environment resets to its initial state, so they change what the policy observes but not the scene's physical configuration.

3.4 DESIGN AND PRE-SPECIFICATION

The grid crosses the three instruction conditions with the three visual levels on 7 tasks and 10 initial states per task, for a total of 630 episodes. The tasks (9, 18, 27, 30, 36, 37, and 73) were fixed in the job-generation script before the grid was run and span high (9, 30, 37), medium (18, 36), and low (27, 73) clean success. All 630 grid episodes ran on the M5 Pro.

Each episode corresponds to one line in a job file that was written before the episode ran. Its seed is the SHA-1 hash of the job fields (experiment, model, task, initial state, instruction condition, visual level, and repetition), so any episode can be rerun from its job line alone. Results are written to

append-only logs keyed by job ID, so interrupted runs can resume without duplicating or dropping episodes. Since the experiment name is part of the hash, episodes from different experiments receive different seeds even if they share a task, an initial state, and an instruction. Section 5 uses this to measure the seed's effect alone. One exception: the environment constructor reseeds with 0, so the first episode of each task in a worker process starts from seed 0 regardless of its job seed, and an exact rerun also needs the episode's position in its worker.

We wrote the analysis plan alongside the job files. The plan's one change came after the grid finished, when maximum-likelihood estimation failed because some cells had zero successes (separation). We switched to Firth's penalized likelihood and recorded the switch before fitting the pooled model.

3.5 Analysis

Primary model. We fit a logistic regression of episode success on the instruction condition, the visual level, and their interaction, with a random intercept for each task,

$$\operatorname{logit} \Pr(y_i = 1) = \beta_0 + \beta_L L_i + \beta_V V_i + \beta_{LV} L_i V_i + u_{t(i)}, \qquad u_t \sim \mathcal{N}(0, \sigma^2_{\text{task}}), \tag{1}$$

where L_i indicates a paraphrased instruction and V_i indicates mild visual perturbation. We fit the model on the 2×2 block of original or paraphrased instructions and no or mild visual perturbation ($n = 280$), since this is the only region with room to measure compounding. Mismatched instructions yield no successes at any visual level, and strong visual perturbation yields almost none (Section 4). We use Firth's penalty (Firth, 1993), which keeps the estimates finite under separation, and compute confidence intervals and p-values from the profile penalized likelihood (Heinze & Schemper, 2002). We implement the model in NumPy and SciPy, integrating out the random effect via numerical quadrature. We validated the implementation against a case with a closed-form solution and verified that the quadrature error is below 10^{-6}.

Probability-scale contrast. Since benchmarks compare success rates, we also measure compounding on the probability scale. Let p_{lv} denote the success rate under instruction condition l (0 for original, 1 for paraphrase) and visual level v (0 for none, 1 for mild). We define

$$\Delta = (p_{10} + p_{01}) - (p_{00} + p_{11}) = (p_{00} - p_{11}) - \big[(p_{00} - p_{10}) + (p_{00} - p_{01})\big], \tag{2}$$

i.e., the drop under both perturbations minus the sum of the two individual drops. A positive Δ indicates a super-additive effect, where the joint drop exceeds the sum, and a negative Δ indicates a sub-additive one. Because $p_{11} \geq 0$, Δ can never exceed $\Delta_{\max} = p_{10} + p_{01} - p_{00}$, so the size of the super-additive effect a design can detect is capped by how far the single perturbations leave success above zero. We report $\Delta_{\max}$ alongside the primary interval on Δ. We compute 95% percentile intervals and a two-sided p-value from 10,000 bootstrap resamples of initial states within tasks, resampling the four episodes of an initial state together. Throughout the paper, we report success rates with Wilson 95% intervals (Wilson, 1927).

4 Do Language and Visual Perturbations Compound?

4.1 Visual perturbation dominates, and language is not ignored

Figure 2 reports success pooled over the 7 tasks, and Table 6 in Appendix E gives the same cells with Wilson intervals. Mild lighting and camera changes reduce success with the original instruction from 57.1% (40/70) to 20.0% (14/70). In the primary model, mild visual perturbation has a log-odds effect of -1.88, a roughly 6.5-fold decrease in the odds of success, while paraphrasing has a much smaller effect (-0.27). Strong visual perturbation leaves little to measure, with 0 of 70 successes under the original instruction and 4 of 70 under the paraphrase. This difference amounts to 4 episodes out of 140 and is not significant (two-sided Fisher exact test, $p = 0.12$), and we do not interpret it as paraphrasing helping under visual degradation.

Mismatched instructions yield no successes in any of the 210 episodes, and each of the 21 task-by-visual-level cells is 0/10. MiniVLA therefore does not ignore its instruction. This contrasts with reports that VLAs ignore language (Fei et al., 2026; Zhou et al., 2025), although the comparison has limits. These studies evaluate other models trained on other LIBERO suites, and LIBERO-Plus measures success on the replaced goal rather than on the original task. Our measure shows that

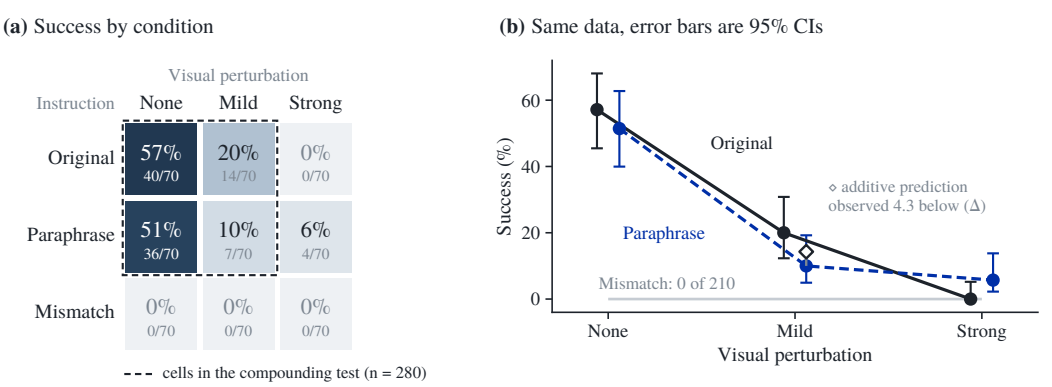

Figure 2: **Visual perturbation drives the drop; paraphrase barely moves it.** (a) Success pooled over 7 tasks, 70 episodes per cell. The dashed box marks the cells used to test for compounding; mismatch and strong visual perturbation sit at or near the floor. (b) The same data, with the original instruction in black and the paraphrase in blue; error bars are Wilson 95% confidence intervals. The diamond is the additive prediction for paraphrase with mild visual perturbation, and its gap to the observed value is Δ.

the instruction of another task prevents the original behavior, but not that the model carries out the new instruction. We also find a much smaller effect of paraphrasing than the 22- to 52-point drops reported by LIBERO-Para (Kim et al., 2026), whose paraphrases systematically vary action and object expressions. Our paraphrases are milder, keeping the object and destination and changing only the verb and phrasing (Appendix A).

4.2 No detectable compounding

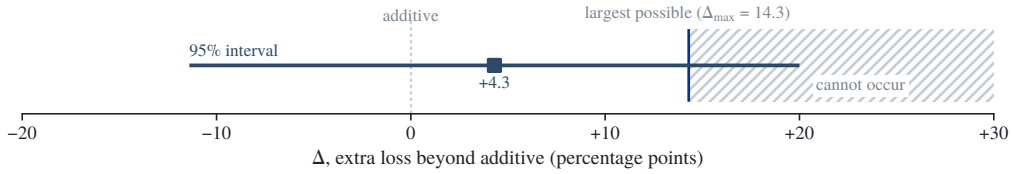

Figure 3: **The design could not have detected a large super-additive loss.** The probability-scale interaction Δ between paraphrase and mild visual perturbation, with its 95% bootstrap interval. Positive values mean the combination does worse than additive. Because success under both perturbations cannot fall below zero, Δ cannot exceed $\Delta_{\max} = 14.3$ points given the observed single-perturbation rates, and the upper end of the interval lies beyond it. The log-odds fits and sensitivity analyses are reported in the text.

In the primary model, the interaction between paraphrase and mild visual perturbation is -0.568 in log-odds (95% CI $[-1.817, 0.637]$, $p = 0.358$), with a standard deviation of 0.822 for the task random intercept. The estimate is stable across specifications. Restricting the fit to the three high-success tasks (9, 30, 37) gives -0.473 ($[-2.441, 1.440]$, $p = 0.627$), and unpenalized maximum likelihood and a Firth fit without the task effect give -0.60 and -0.55, respectively. Fitting the full 3×3 grid gives -0.569 ($[-1.820, 0.638]$, $p = 0.358$), but this agreement is expected rather than informative, because the mismatch and strong-vision cells sit at or near zero and add almost no information about this interaction. In a sensitivity analysis that drops the two low-success tasks (27 and 73), the estimate is -0.936 ($[-2.351, 0.403]$, $p = 0.173$), with a task standard deviation of 0.276. None of these intervals excludes zero.

On the probability scale, $\Delta = +0.043$ (95% CI $[-0.114, 0.200]$, $p = 0.646$; Figure 3), and dropping tasks 27 and 73 gives $\Delta = +0.120$ ($[-0.060, 0.300]$). These intervals must be read against the floor. With $p_{00} = 0.571$ and $p_{01} = 0.200$, the largest achievable value is $\Delta_{\max} = p_{10} - 0.371$, reached

if the joint condition had no successes at all. With $p_{10} = 36/70$, $\Delta_{\max} = 14.3$ points. The upper end of the primary interval (20 points) lies above anything the design could have produced, so the probability-scale result rules out no achievable super-additive loss, and the log-odds interaction is the informative test. The lower end of the primary interval admits sub-additivity of up to about 11 points. We therefore report no detectable compounding rather than an absence of compounding. The data are consistent with additive effects, but also with a moderate interaction in either direction, and a stronger test needs perturbation strengths that leave more room above zero success.

Both scales point the same way. The log-odds interaction is negative and Δ is positive, but by their definitions both mean that the joint condition does somewhat worse than additivity predicts. Appendix C explains why we report both.

5 Are Episode Outcomes Reproducible?

The seed alone changes a quarter of outcomes. Our design contains a direct test of reproducibility. The clean-baseline run and the grid cell with the original instruction and no visual perturbation include the same 70 episodes in every respect the benchmark defines: the same 7 tasks and 10 initial states, the same instruction, greedy decoding, and MPS in fp32 on the same machine. The only difference is the seed, which is derived from the experiment name. Without visual perturbation and under greedy decoding, the seed sets only the global PyTorch, NumPy, and Python random states, and our rollout code draws no random numbers from them; so if an episode were determined by its task, initial state, and instruction, the two sets would be identical.

They are not. Of the 70 pairs, 18 change outcome (25.7%, Wilson 95% CI $[16.9, 37.0]$), and 44 (63%) end at a different step (Figure 1a). Aggregate success barely moves, from 38 to 40 of 70 (Figure 1b), because changes in the two directions nearly cancel. An evaluation that reports only aggregate success would see two nearly identical runs, while about a quarter of the individual outcomes differ.

The seed reaches the episode through LIBERO, which redraws the positions of scene fixtures such as the stove on every reset from NumPy's global random state; initial states do not record these positions. This is not a defect of MiniVLA, rather one of evaluation and reporting. Benchmarks define an episode by its task, initial state, and instruction, and the simulator's remaining random state is neither controlled nor reported, yet it changes about a quarter of individual outcomes. We have checked same-seed reproducibility for only one episode. On the M5 Pro, rerunning task 9 from initial state 0 reproduced the original episode exactly, down to its length of 118 steps.

The backend appears to matter as much (preliminary). Early in the project, we ran the 80 clean episodes on the M4 Pro with MPS and then on CPU, both in fp32 and with identical job IDs. The CPU run was stopped early because some episodes hung inside native PyTorch or MuJoCo code.[1] 34 pairs completed. Of these, 7 episodes changed outcome between the two backends across four tasks (9, 18, 27, and 30), five in favor of CPU, and every outcome change also changed the episode length. For these 7 episodes, the M5 Pro run for the same job IDs matched the MPS outcome and length in 6. With only 34 pairs, we treat this as an indication that switching the backend at fixed fp32 precision perturbs episodes by an amount comparable to changing the seed. In LLM decoding, fp32 nearly eliminates divergence across GPU configurations (Yuan et al., 2025). Whether the same holds in closed-loop control remains open.

We have not measured the mechanism directly, but one account fits the backend result. The policy makes seven argmax decisions per step over hundreds of steps, so even rare near-ties between the top two bins will flip some action, moving it by one bin ($2/256$ in normalized units). Contact-rich manipulation can then amplify the resulting state difference until the outcome changes. A different seed instead moves the scene fixtures, and a different backend perturbs the logits through different kernels, accumulation orders, and possibly preprocessing.

For evaluation, this means that at a fixed initial state the outcome of an episode behaves as a random variable rather than a fixed quantity, with two consequences. First, two conditions are not paired just because they share initial states. In our grid, as in most benchmarks, each condition runs with its

[1] Python's `SIGALRM`-based timeout cannot interrupt a call that is blocked in native code.

own seed, so about a quarter of the per-episode differences between any two conditions could arise with no change in condition. Conditions should therefore be compared through aggregate rates with interval estimates over many initial states and, ideally, several seeds, not episode by episode. Second, rerunning published seeds on a different backend or software environment should not reproduce the same episodes, so the backend belongs to the reported experimental condition.

6 How Much Variability Does Sampling Add?

Stochastic decoding is a simple way to obtain different rollouts from the same initial state, for example, to estimate the probability of success from that state. We ask how the variability it produces compares with a noise control of known scale. We use tasks 18 and 36 with 5 initial states, 10 repetitions, and three conditions, for a total of 300 episodes on the M3 (MPS, fp32), all of which completed without errors. In the sampled condition, the policy samples its action tokens with the default generation settings of transformers 4.40.1, since neither our code nor the checkpoint sets them (temperature 1.0, top-k 50, no top-p truncation). In the two noise conditions, the policy decodes greedily, and we add Gaussian noise to the six arm dimensions in the normalized action space before unnormalization, with a standard deviation of one or three bin widths ($\sigma = k \cdot 2/256$, $k \in \{1, 3\}$). The gripper dimension is not perturbed, and the noise is seeded from the job seed. We tied the noise scale to the action tokenizer bin width and fixed it in the analysis plan before collecting any diversity episodes.

Success counts are similar across the three conditions (Table 5 in Appendix D), and we do not test the differences, since the experiment was designed to compare variability rather than success. The mean pairwise per-step action distance does not differ between sampling and either noise level, so sampling perturbs a rollout by about one to three bin widths of Gaussian action noise. Counting distinct full trajectories does not discriminate, since any single-bin difference over hundreds of steps makes a trajectory unique, and all three conditions reach 10 of 10. We therefore count distinct first actions, binned to the 256-bin action grid, among the 10 repetitions from each initial state. Both noise conditions reach the maximum of 10, while sampling yields 8.00 on average because the sampled first action often repeats. For this model, then, sampling is not a richer source of behavioral variation than a noise control whose scale the experimenter sets explicitly.

7 Discussion and Recommendations

Additivity is not contradicted for the one pair of perturbations we could test, though floor effects limit how strong a test this is. Reproducibility does not hold: changing only the seed changes about one in four episode outcomes on the same machine and backend, and our preliminary data suggest the backend has a similar effect. Aggregate success stays stable even when individual outcomes change, so none of this makes aggregate benchmark numbers wrong, but their validity depends on conditions that are rarely tested or reported (Table 1).

Fragile outcomes matter more as compact models make it practical to evaluate on CPUs, laptops, and consumer GPUs, which adds the backend to the incidental details that already change individual episodes. Our null result on compounding does not contradict the interactions LIBERO-Plus reports among visual and physical factors (Fei et al., 2025), since interactions may depend on the perturbation pair and the model. We make the following recommendations for small-sample VLA robustness evaluations.

1. Report seeds, inference backend, precision, hardware, and simulator version as part of the experimental condition, and hold them fixed within paired comparisons.
2. Treat the outcome at a fixed initial state as random: spread episodes over many initial states and seeds, and report interval estimates rather than per-episode differences.
3. Fix episodes, seeds, and the analysis plan before running experiments, and keep per-episode logs.
4. Test perturbations jointly, report the interaction with an interval and the largest value the marginals allow, and choose strengths that leave headroom above zero success.
5. When diverse rollouts are needed, compare sampling against a noise control of known scale.

Table 1: What we tested, what we found, and what it implies for evaluation. pp: percentage points.

Question	Test	Answer	Recommendation
Do perturbations compound?	Instruction × vision factorial, 7 tasks, 630 episodes	**Not detectably.** Floor effects cap detectable super-additive loss at 14.3 pp.	Test perturbations jointly at strengths with headroom, and report the interaction with an interval and its ceiling.
Does the model use its instruction?	Instruction of a different task	**Yes.** 0 of 210 episodes succeed on the original task.	Include a mismatched-instruction control.
Is an episode fixed by its task, initial state, and instruction?	Same 70 episodes on one machine and backend, only the seed differs	**No.** 18 of 70 outcomes change (95% CI 17% to 37%); aggregate success 38 vs. 40 of 70.	Repeat episodes over seeds and compare aggregate rates with intervals, not individual episodes.
Does the inference backend matter?	Clean episodes, MPS vs. CPU, fp32 (preliminary)	**Likely.** 7 of 34 outcomes change.	Report the backend and hold it fixed within comparisons.
Does sampling add more variability than noise?	Sampling vs. greedy plus Gaussian noise, 300 episodes	**No.** Per-step action variability matches 1 to 3 bins of noise.	Compare sampling with a noise control of known scale.

8 LIMITATIONS

We study one model in one simulator, and all experiments ran on Apple silicon. None use CUDA, where most published VLA numbers are computed, so we do not know how large the seed and backend effects are on NVIDIA GPUs. Our backend result is also preliminary, and we verified exact reproduction under the same seed for a single episode. The machines also differ in macOS version, and the M3 used an older PyTorch, so the software environment was not identical across machines.

Floor effects restrict the compounding test to paraphrase and mild visual perturbation, and even there they cap the super-additive effect the probability-scale contrast can detect (Section 4). Mismatched instructions yield zero successes (0/210) and strong visual perturbation yields almost none (0/70 and 4/70), so we cannot say whether stronger perturbations compound. With seven tasks, the model includes only a random intercept per task, so we cannot reliably estimate a task-specific interaction. Given the seed result, cell-level differences of a few episodes should not be read as effects.

We have not cross-checked our Firth mixed model against R implementations such as `logistf`. Finally, pre-specification relied on our own records rather than a public registry, and the switch to Firth's penalty followed the grid's zero-success cells, although it preceded the pooled fit.

AI USE STATEMENT

In this work, we used generative AI tools (Claude, Anthropic) to implement methods, including experiment code, job orchestration, and analysis scripts, to give feedback on methodology and experimental design, and to help interpret results. We also used them to search for and summarize related literature and to draft and polish text from the authors' notes and verified results. We did not use generative AI tools to formulate mathematical claims or proofs or to generate synthetic data sets. Translation, dataset cleaning, and qualitative data analysis are not applicable to this work. We have reviewed all AI-assisted work and take responsibility for the final content of this work.

ETHICS STATEMENT

This work uses only simulation and involves no human subjects, personal data, or physical robots. Its goal is to make robustness claims about robot policies easier to verify and reproduce.

Reproducibility statement

Section 3 specifies the checkpoint, simulator and library versions, numerical precision, backends, and hardware, and Appendix A lists the instructions and perturbation parameters. Each episode is defined by one line of a job file, and its seed is the hash of that line's fields, so we can rerun every episode in this paper from the job files. Exact per-episode reproduction additionally requires the same backend and software environment, and we verified it for one episode (Section 5). For review, we provide the rollout and analysis code at `https://anonymous.4open.science/r/compact-vla-evals`. The job files, analysis plan, and per-episode logs will be released with the camera-ready version. All experiments ran on single Apple silicon machines. One MPS episode takes about 108 s on the M5 Pro, the 300 diversity episodes took 11.6 hours of wall-clock time on the M3, and the longest grid episode on the M5 Pro took 486 s against a 600 s timeout.

References

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Chunrong Ai and Edward C. Norton. Interaction terms in logit and probit models. *Economics Letters*, 80(1):123–129, 2003.

Suneel Belkhale and Dorsa Sadigh. MiniVLA: A better VLA with a smaller footprint. Stanford AI Lab Blog, `https://ai.stanford.edu/blog/minivla/`, 2024.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning (CoRL)*, 2023.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep RL: A case study on PPO and TRPO. In *International Conference on Learning Representations (ICLR)*, 2020.

Yu Fang, Yuchun Feng, Dong Jing, Jiaqi Liu, Yue Yang, Zhenyu Wei, Daniel Szafir, and Mingyu Ding. When vision overrides language: Evaluating and mitigating counterfactual failures in VLAs. *arXiv preprint arXiv:2602.17659*, 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: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

David Firth. Bias reduction of maximum likelihood estimates. *Biometrika*, 80(1):27–38, 1993.

Jianing Guo, Zhenhong Wu, Chang Tu, Yiyao Ma, Xiangqi Kong, Zhiqian Liu, Jiaming Ji, Shuning Zhang, Yuanpei Chen, Kai Chen, Qi Dou, Yaodong Yang, Xianglong Liu, Huijie Zhao, Weifeng Lv, and Simin Li. On robustness of vision-language-action model against multi-modal perturbations. In *International Conference on Learning Representations (ICLR)*, 2026.

Horace He and Thinking Machines Lab. Defeating nondeterminism in LLM inference. Thinking Machines Lab: Connectionism, `https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/`, 2025.

Georg Heinze and Michael Schemper. A solution to the problem of separation in logistic regression. *Statistics in Medicine*, 21(16):2409–2419, 2002.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *AAAI Conference on Artificial Intelligence*, 2018.

Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic VLMs: Investigating the design space of visually-conditioned language models. In *International Conference on Machine Learning (ICML)*, 2024.

Chanyoung Kim, Minwoo Kim, Minseok Kang, Hyunwoo Kim, and Dahuin Jung. LIBERO-Para: A diagnostic benchmark and metrics for paraphrase robustness in VLA models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2026.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, 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 *Conference on Robot Learning (CoRL)*, 2024.

Hadas Kress-Gazit, Kunimatsu Hashimoto, Naveen Kuppuswamy, Paarth Shah, Phoebe Horgan, Gordon Richardson, Siyuan Feng, and Benjamin Burchfiel. Robot learning as an empirical science: Best practices for policy evaluation. *arXiv preprint arXiv:2409.09491*, 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 (NeurIPS), Datasets and Benchmarks Track*, 2023.

Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are GANs created equal? A large-scale study. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How I learned to start worrying about prompt formatting. In *International Conference on Learning Representations (ICLR)*, 2024.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

David Snyder, Asher James Hancock, Apurva Badithela, Emma Dixon, Patrick Miller, Rares Andrei Ambrus, Anirudha Majumdar, Masha Itkina, and Haruki Nishimura. Is your imitation learning policy better than mine? Policy comparison with near-optimal stopping. In *Robotics: Science and Systems (RSS)*, 2025.

Cecilia Summers and Michael J. Dinneen. Nondeterminism and instability in neural network optimization. In *International Conference on Machine Learning (ICML)*, 2021.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5026–5033, 2012.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927.

Jiayi Yuan, Hao Li, Xinheng Ding, Wenya Xie, Yu-Jhe Li, Wentian Zhao, Kun Wan, Jing Shi, Xia Hu, and Zirui Liu. Understanding and mitigating numerical sources of nondeterminism in LLM inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Borong Zhang, Jiahao Li, Jiachen Shen, Yuhao Zhang, Yishuai Cai, Hailu Ji, Yuanpei Chen, Juntao Dai, Jiaming Ji, and Yaodong Yang. VLA-Arena: An open-source framework for benchmarking vision-language-action models. In *International Conference on Machine Learning (ICML)*, 2026.

Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. LIBERO-PRO: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.

Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, and Yifeng Zhu. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020.

Table 2: Instructions for the seven grid tasks.

Task	Original	Paraphrase	Mismatch
9	put the black bowl on the plate	move the black bowl onto the plate	pick up the book and place it in the front compartment of the caddy
18	put the frying pan on the stove	set the frying pan down on the stove	put the wine bottle on the wine rack
27	put the wine bottle on the wine rack	stand the wine bottle up on the wine rack	pick up the alphabet soup and put it in the basket
30	put the black bowl on the plate	transfer the black bowl onto the plate	put the frying pan on the stove
36	put the white bowl on the plate	move the white bowl onto the plate	put the wine bottle on the wine rack
37	put the white bowl to the right of the plate	place the white bowl at the right side of the plate	put the frying pan on the stove
73	pick up the book and place it in the front compartment of the caddy	lift the book and set it into the front compartment of the caddy	put the white bowl on the plate

A Instructions and perturbation parameters

Table 2 lists the instructions. Originals are read at runtime from the installed LIBERO benchmark. Tasks 9 and 30 share the same original instruction but received different paraphrases. Table 3 lists the visual perturbation parameters.

Table 3: Visual perturbation levels. Light scale multiplies the diffuse and ambient lighting. Camera jitter is the standard deviation of a Gaussian offset added to each coordinate of the third-person camera position, drawn once per episode from its seed. Both are applied after the environment is reset to its initial state.

Level	Light scale	Camera jitter (m)
None	1.0	0.00
Mild	0.5	0.03
Strong	0.25	0.08

B Clean success per task

Table 4: Clean success per task with MPS in fp32 on the M5 Pro (10 episodes per task, one seed each).

Task	9	18	27	30	36	37	46	73	Total
Successes	8	5	1	8	5	9	1	2	39/80

C Log-odds and probability-scale interactions

Our two estimates have opposite arithmetic signs only because of their definitions, and they agree in direction. In general, however, the two scales need not agree even in direction. Effects that are additive on the probability scale interact on the log-odds scale, and vice versa; near a floor, the two interactions can even point in opposite directions (Ai & Norton, 2003). We report both because they answer different questions. The log-odds model accounts for differences in task difficulty, handles separation, and is not capped by the floor in the same way, whereas Δ is measured on the scale used to compare benchmark results.

D Sampling and noise results

E Grid results

Table 5: Sampling adds about as much per-step action variability as small Gaussian action noise. Tasks 18 and 36, 5 initial states $\times$ 10 repetitions per condition (300 episodes). Noise conditions decode greedily and add Gaussian noise of k bin widths to the six arm dimensions. Distinct first actions are counted among the 10 repetitions from each initial state after binning to the 256-bin action grid. Differences are relative to sampling, with percentile bootstrap 95% intervals (10,000 resamples) over the 10 (task, initial state) blocks, resampling the three conditions of a block together. With only 10 blocks, these intervals are rough.

	Success		Distinct first actions (of 10)		Pairwise action distance
Condition	Task 18	Task 36	Mean	Difference from sampling	Difference from sampling
Sampling	28/50	26/50	8.00	ref.	ref.
Noise, $k = 1$	26/50	22/50	10.00	$+2.00$ [$+1.00$, $+3.20$]	-0.0135 [-0.0508, $+0.0255$]
Noise, $k = 3$	27/50	24/50	10.00	$+2.00$ [$+0.90$, $+3.20$]	-0.0100 [-0.0320, $+0.0105$]

Table 6: Success pooled over 7 tasks (70 episodes per cell), with Wilson 95% intervals. The compounding analysis uses the upper-left 2×2 block.

Instruction	No visual perturbation	Mild	Strong
Original	40/70 (57.1%) [45.5, 68.1]	14/70 (20.0%) [12.3, 30.8]	0/70 (0.0%) [0.0, 5.2]
Paraphrase	36/70 (51.4%) [40.0, 62.8]	7/70 (10.0%) [4.9, 19.2]	4/70 (5.7%) [2.2, 13.8]
Mismatch	0/70 (0.0%) [0.0, 5.2]	0/70 (0.0%) [0.0, 5.2]	0/70 (0.0%) [0.0, 5.2]