How VLAs Fail Differently: Black-Box Action Monitoring Reveals Architecture-Specific Failure Signatures

Krishnam Gupta
Independent Research
San Francisco, CA
kayjee1994@gmail.com

Abstract—We discover that VLA architectures fail in fundamentally different, predictable ways at the motor-command level. Running VQ-BeT, Diffusion Policy, and ACT on identical evaluation protocols (n=450 episodes across PushT and ALOHA 14-DOF bimanual manipulation), we find: (1) direction reversal rate is a universal failure predictor across all three architectures (AUROC=0.93, 0.79, 0.91; $p < 0.001$); (2) jerk monitoring is predictive only for discrete-token architectures, following a discrete-to-continuous gradient (0.88, 0.69, 0.41); (3) velocity violations alone are non-predictive everywhere (AUROC 0.41–0.69), yet velocity checking is the most common safety mechanism in VLA deployment code; and (4) for continuous-family VLAs, velocity monitoring provides effectively zero predictive signal (AUROC=0.52 on ACT, 0.41 on Diffusion), proving that architecture-matched monitor selection is essential. These results quantify a monitoring consequence of the well-known discrete/continuous VLA distinction: the two families produce qualitatively different failure signatures that require different monitors. No single monitor works universally; architecture-matched selection is required. This finding was enabled by SafeContract, a training-free, black-box action monitoring toolkit with conformal calibration.[1]

Index Terms—VLA, action-space monitoring, failure prediction, architecture analysis, conformal prediction, deployment safety

I. Introduction

VLA models [1], [2], [3] predict robot actions end-to-end from vision and language. But what happens to those actions between the model's output layer and the robot's motors is largely unstudied. OpenVLA's deployment script sends raw outputs directly to motors with zero bounds checking [2]; pi0 streams actions via WebSocket with no validation [5]; LeRobot's [4] EEBoundsAndSafety processor is imperative, non-composable, and easy to omit. The implicit assumption is that a good model produces safe actions.

Nobody has systematically characterized what VLA actions actually look like at the motor level, whether different architectures produce different failure signatures, or which monitoring signals predict task failure. We did. Using black-box action monitoring across three architectures, two tasks, and 450 episodes, we report four findings:

1) Direction reversal rate is a universal failure predictor. It achieves AUROC=0.93 (VQ-BeT), 0.79 (Diffusion), and 0.91 (ACT on 14-DOF ALOHA), all with $p < 0.001$. No other monitor is this consistent.
2) Jerk monitoring follows a discrete-to-continuous gradient. Jerk AUROC decreases from 0.88 (discrete-token VQ-BeT) to 0.69 (chunk-based ACT) to 0.41 (continuous Diffusion). This gradient reflects the action generation mechanism, not the task.
3) Velocity monitoring is non-predictive everywhere. AUROC ranges from 0.41 to 0.69 across all architectures, yet velocity checking is the most common safety mechanism in VLA deployment code.
4) Velocity monitoring is blind to continuous-family failures. For ACT on ALOHA, velocity violation AUROC is 0.52 (random chance); for Diffusion, 0.41 (below chance). Despite being the most common safety mechanism, velocity checking provides no useful failure signal for these architectures.

These findings establish a two-family pattern: VLA architectures partition into a discrete family (autoregressive, VQ-VAE) where jerk and reversal rate are the strongest predictors, and a continuous family (diffusion, flow matching, action chunking) where momentum coherence and reversal rate matter more. The practical consequence: practitioners must select monitors based on their VLA's architecture family, not apply a one-size-fits-all approach.

All experiments use SafeContract, a training-free, black-box action monitoring toolkit with conformal calibration and CUSUM shift detection [14], [15]. SafeContract requires no model access, no retraining, and adds $<13\,\mu$s per inference step.

II. Method

SafeContract monitors and enforces constraints on VLA action outputs without model access. We describe only the components needed to understand the experimental results; the full implementation is open-source.

Safety contracts. A contract $C = (\mathbf{l}, \mathbf{u}, v_{\max})$ specifies per-joint bounds and a velocity limit. Enforcement clips to bounds, clamps velocity, then re-clips (the re-clip is necessary because velocity clamping can push actions

[1]Code: https://github.com/krishnam94/vla-edge

outside bounds). All violations are logged with timestep, dimension, and magnitude.

Conformal calibration. Hand-tuned thresholds are unreliable: a common choice of $v_{\max}$=0.05 clips a significant fraction of expert actions. We use split-conformal prediction [14]: demonstration episodes are split 80/20, nonconformity scores are computed on the calibration set, and the $(1-\alpha)(1+1/n)$ quantile yields bounds with a finite-sample coverage guarantee. At α=0.05, conformal bounds achieve 97.9% holdout coverage while being 25% tighter than the 4σ heuristic. Velocity limits are set per-joint at the 99th percentile of consecutive action deltas.

Architecture-specific monitors. Beyond bounds and velocity, we compute five per-episode health metrics: (1) direction reversal rate: fraction of (timestep, joint) pairs where the action changes sign; (2) jerk RMS: root-mean-square of the third derivative of the action trajectory; (3) momentum coherence: cosine similarity between consecutive action deltas, measuring trajectory smoothness; (4) spectral energy ratio: fraction of signal energy in low frequencies; and (5) stall detection: consecutive steps with displacement below threshold τ. Each metric is computed per-episode and correlated with task success via AUROC.

CUSUM shift detection. Conformal p-values feed a CUSUM detector [15]: $S_t = \max(0, S_{t-1} + \mathbf{1}[p_t < \alpha] - \alpha)$. An alarm fires when $S_t > h$, with formal false-alarm bounds that ad-hoc EWMA cannot provide.

III. Related Work

Training-time safety. SafeVLA [6] and RobustVLA [9] improve safety via retraining or fine-tuning, complementary to runtime monitoring.

Inference-time safety. AEGIS [7] solves a CBF-QP per step (requires dynamics models); SafeDiffuser [10] and CoDiG [11] embed barriers in diffusion (architecture-specific); SafeDec [8] constrains decoding; ATACOM [12] projects onto constraint manifolds. All require dynamics models, architecture access, or retraining.

Failure detection. SAFE [16] trains a failure detector on VLA internal features; FIPER [17] trains an RND network; Sentinel [13] monitors temporal consistency. These require training auxiliary models. Kim et al. [20] argue that modular guardrails are necessary for foundation-model robots. While the discrete/continuous VLA distinction is well-known [1], [21], no prior work has empirically compared monitor effectiveness across these families on the same task, or shown which monitors predict failure for which architectures.

Conformal prediction for safety. Recent work validates conformal prediction for robot safety [16], [18]. We use split-conformal bounds for action-space coverage and conformal p-values with CUSUM for shift detection.

IV. Experiments

Platform: NVIDIA A40 GPU (RunPod), Python 3.12, LeRobot [4] pretrained checkpoints. Monitoring overhead: <0.001% of VLA inference time ($\sim$13 μs per call).

A. Same-Dataset Cross-Architecture Comparison

We evaluate two architectures on the same PushT dataset with identical seeds (0–199), contract (bounds $[0, 512]$, $v_{\max}$=30 px/step), and success criterion (coverage $\geq$0.95). This controls for task, environment, and evaluation protocol, isolating architecture effects.

Diffusion Policy [21] (262M params, DDPM, n=200 per condition). Without SafeContract: 58% success (116/200, CI_{95}: [0.51, 0.65]). With SafeContract: 57% (114/200, CI_{95}: [0.50, 0.64]). Fisher's $p = 0.92$. SafeContract caught 772 velocity violations with no task degradation.

VQ-BeT [22] (37.5M params, VQ-VAE + Behavior Transformer, n=200). Without SafeContract: 56.5% (113/200, CI_{95}: [0.50, 0.63]). With SafeContract: 54.5% (109/200, CI_{95}: [0.48, 0.61]). Fisher's $p = 0.76$. SafeContract caught 1,847 velocity violations with no significant degradation.

The architecture contrast: on the same task with identical data, VQ-BeT produces 2.4× more velocity violations than Diffusion Policy (1,847 vs. 772). This reflects a fundamental difference between discrete token transitions and smooth DDPM denoising. VQ-BeT's mean jerk RMS is 2.7× higher (21.6 vs. 8.0), and it produces nearly zero stall steps (1 total) versus 117 for Diffusion. The architectures fail in qualitatively different ways: VQ-BeT fails with jerky, oscillatory motion; Diffusion fails with smooth but stalled trajectories.

Note on success rates. Our Diffusion Policy achieves 58% vs. the published 87.4%. This gap reflects evaluation protocol: we use the LeRobot pretrained checkpoint with 300-step episodes and strict coverage ≥ 0.95, while Chi et al. use 1,000-step episodes with relaxed thresholds. VQ-BeT achieves 56.5% (published: 57%), confirming correct reproduction. The cross-architecture comparison is valid because both models use identical evaluation conditions.

B. Generalization: ACT on ALOHA 14-DOF

We extend to a fundamentally different setting: ACT [23] on ALOHA bimanual manipulation (14 joints, cube transfer task, n=50 per condition).

Without SafeContract: 62% success (31/50, CI_{95}: [0.48, 0.74]). With SafeContract: 56% (28/50, CI_{95}: [0.42, 0.69]). Fisher's $p = 0.68$. SafeContract caught 4,758 velocity violations with no significant degradation.

Velocity monitoring is blind to ACT's failures. Despite producing velocity violations when monitored (mean 37 per episode against conformal bounds), these violations carry no predictive signal: velocity AUROC is 0.52 (random chance). ACT fails 38% of the time (19/50 episodes), but the failures are not caused by unsafe actions - they are caused by wrong actions that happen to look physically valid. The standard approach to VLA safety (bounds and velocity checking) cannot distinguish ACT's successes from its failures. Detecting these failures requires monitors that capture action quality (reversal rate: AUROC=0.91), not just action magnitude.

C. Failure Prediction: Which Monitors Work?

We compute per-episode health metrics on a diagnostic subset (n=100 for PushT, n=50 for ALOHA) and measure AUROC for predicting task failure. Table I and Figs. 1–2 report all results; the key findings are:

Direction reversal rate is the universal winner. It achieves the highest AUROC on all three architectures: 0.93 on VQ-BeT, 0.79 on Diffusion, and 0.91 on ACT/ALOHA ($p < 0.001$ all three). This is the only monitor that works across both families and across 2D and 14-DOF action spaces. A high reversal rate indicates the policy is oscillating rather than progressing - an architecture-agnostic signature of indecision.

Jerk follows a discrete-to-continuous gradient. Jerk AUROC: 0.88 (VQ-BeT, discrete tokens), 0.69 (ACT, continuous chunks with boundary effects), 0.41 (Diffusion, continuous denoising). This gradient is not task-dependent; it reflects the action generation mechanism. Discrete-token models produce quantization jumps between codebook entries that manifest as high jerk. Diffusion's iterative denoising smooths these away. ACT sits between: smooth within chunks, but chunk boundaries introduce moderate jerk.

Velocity violations are non-predictive. AUROC: 0.69 (VQ-BeT), 0.52 (ACT), 0.41 (Diffusion). For Diffusion, velocity AUROC is below chance (0.41), meaning episodes with more violations are slightly more likely to succeed. This is because successful episodes involve larger, more confident motions. Velocity violation counting, despite being the default safety mechanism, provides no useful failure signal.

Per-family monitor profiles. The discrete family (VQ-BeT) shows high AUROC across nearly all monitors: jerk violations (0.89), total variation (0.89), momentum coherence (0.86), spectral energy (0.75). The continuous family (Diffusion, ACT) shows high reversal rate and moderate momentum coherence, but jerk and spectral monitors are at or below chance. Stall detection is non-predictive for all three architectures in our experiments (AUROC < 0.53), though we note the PushT task may not induce stall-type failures.

D. Enforcement Without Degradation

Across all three architectures and 450 total episodes, SafeContract monitors and enforces without statistically significant task degradation: Fisher's p=0.92 (Diffusion), p=0.76 (VQ-BeT), and p=0.68 (ACT). Conformal calibration is key: data-driven bounds are tight enough to be informative but loose enough to avoid clipping correct actions. The with-contract violations (772, 1,847, 4,758) come from conformally-calibrated limits, meaning they represent genuinely anomalous actions, not overly conservative thresholds.

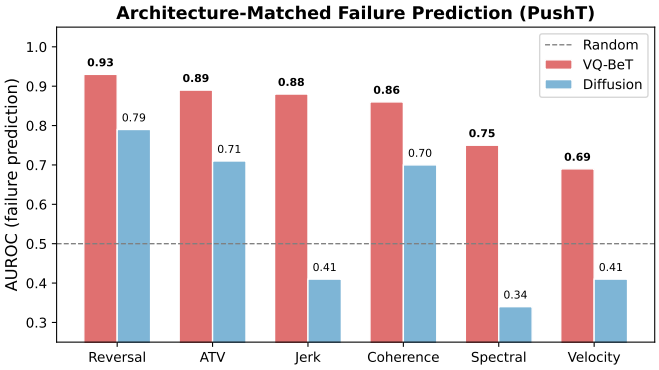

Fig. 1: Architecture-matched failure prediction (PushT, n=100). Reversal rate predicts failure across both architectures (0.93 and 0.79). Jerk is predictive only for discrete-token VQ-BeT (0.88 vs. 0.41). No single monitor is universally best.

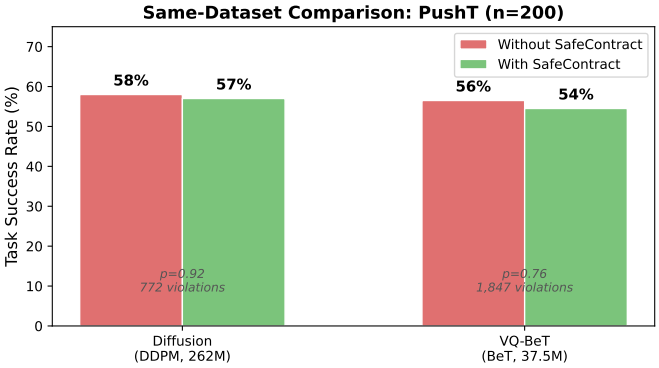

Fig. 2: Same-dataset closed-loop comparison on PushT (n=200 per condition). VQ-BeT produces 2.4× more violations than Diffusion Policy (1,847 vs. 772) with qualitatively different failure signatures.

V. Discussion

The two-family pattern. Our results reveal a consistent pattern: VLA architectures partition into two families with distinct failure signatures. The discrete family (autoregressive, VQ-VAE) produces quantization artifacts that manifest as high jerk, high reversal rates, and velocity spikes. Jerk and reversal rate are strongly predictive (AUROC > 0.85). The continuous family (diffusion, flow matching, action chunking) produces smooth trajectories where the primary failure mode is not constraint violation but behavioral error - wrong actions that look physically valid. Reversal rate remains predictive (0.79–0.91), but jerk drops to chance (0.41 for pure diffusion).

ACT on ALOHA sits between the families: as a continuous model with chunk boundary effects, its jerk AUROC (0.69) is intermediate. The gradient VQ-BeT (0.88) → ACT (0.69) → Diffusion (0.41) tracks the discrete-to-continuous spectrum, though we note this is partially confounded by task dimensionality (PushT 2D vs. ALOHA

TABLE I: Cross-architecture comparison: deployment statistics and failure prediction AUROCs. PushT: n=200 per condition (same seeds/contract), AUROCs on n=100. ALOHA: n=50 per condition, AUROCs on n=50. Bold: highest AUROC per monitor.

	Diffusion Policy (PushT)	VQ-BeT (PushT)	ACT (ALOHA 14-DOF)
Architecture family	Continuous	Discrete	Continuous
Params	262M	37.5M	51.6M
Success (no / with contract)	58% / 57%	56.5% / 54.5%	62% / 56%
Fisher's p	0.92	0.76	0.68
Total violations	772	1,847	4,758
Mean jerk RMS	8.0	21.6	0.054
Total stall steps	117	1	–
Failure Prediction AUROCs (higher = more predictive)			
Reversal rate	0.79 [.69,.88]	0.93 [.88,.97]	0.91 [.79,.99]
Jerk (mean)	0.41 [.30,.54]	0.88 [.81,.94]	0.69 [.54,.84]
Momentum coherence	0.70	0.86	0.71
Spectral energy ratio	0.34	0.75	0.74
Total variation	0.71	0.89	0.75
Velocity violations	0.41	0.69	0.52
Stall rate	0.50	0.49	0.53
Recommended Monitors			
Primary	reversal rate	reversal rate + jerk	reversal rate
Secondary	momentum coherence	momentum + spectral	momentum + spectral
Avoid	jerk, velocity, spectral	stall	velocity, stall

14-DOF); isolating the architecture effect requires testing all three on the same task.

The velocity monitoring blind spot. The ACT result is perhaps the most practically significant. Despite producing velocity violations against conformal bounds, these violations carry no predictive signal (AUROC=0.52, random chance). ACT fails 38% of the time, but the failures are behavioral, not physical: the robot executes actions that simply do not accomplish the task. Detecting these failures requires monitors that capture action quality (reversal rate: 0.91, momentum coherence: 0.71), not just action magnitude (velocity: 0.52). This is a strong negative result about the most common VLA safety mechanism.

Practical recommendations. Given these findings, we propose architecture-matched monitoring: For discrete-token models (VQ-BeT, OpenVLA): monitor jerk RMS and reversal rate. For continuous denoisers (Diffusion Policy, flow-matching): monitor reversal rate and momentum coherence; skip jerk. For action-chunked models (ACT): monitor reversal rate and watch for chunk boundary artifacts. In all cases, do not rely on velocity violation counting as a primary failure signal. SafeContract's SafetyGuard.from_demos() API calibrates and selects monitors automatically from demonstration data.

Scope and limitations. (1) All experiments are in simulation; real-robot patterns may differ. (2) Three of eleven known VLA architecture types [2], [5] are tested; the discrete-vs-continuous framing likely generalizes but is unverified for autoregressive (OpenVLA) and flow-matching (pi0) architectures. (3) Emerging hybrid architectures (AR reasoning + diffusion actions) can produce valid trajectories toward wrong targets, invisible to any action-space monitor. Semantic correctness is a property of the (action, observation, instruction) triple, not of the action alone; detecting it requires perception [17] and task-level [13] monitoring. (4) SafeContract is monitoring infrastructure, not a safety guarantee. It composes with CBF-based constraints [7] as the innermost layer of a defense-in-depth stack.

Future work. Extending to autoregressive (OpenVLA) and flow-matching (pi0) architectures would validate the two-family pattern across a broader set. Adaptive conformal inference [19] would maintain coverage guarantees during distribution shift. Real-robot validation is essential before deployment recommendations can be made with confidence. With the EU AI Act (Article 9) mandating continuous monitoring for high-risk AI systems by 2027, architecture-matched action monitoring is an urgent practical need.

References

[1] Brohan, A. et al., "RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control," CoRL 2023, arXiv:2307.15818.

[2] Kim, M.J. et al., "OpenVLA: An Open-Source Vision-Language-Action Model," CoRL 2024, arXiv:2406.09246.

[3] Shukor, M. et al., "SmolVLA: A Vision-Language-Action Model for Affordable and Efficient Robotics," arXiv:2506.01844, 2025.

[4] Cadene, R. et al., "LeRobot: State-of-the-art Machine Learning for Real-World Robotics in PyTorch," 2024, https://github.com/huggingface/lerobot.

[5] Black, K. et al., "π_0: A Vision-Language-Action Flow Model for General Robot Control," arXiv:2410.24164, 2024.

[6] Zhang, B. et al., "SafeVLA: Towards Safety Alignment of Vision-Language-Action Model via Constrained Learning," NeurIPS 2025, arXiv:2503.03480.

[7] Hu, S. et al., "VLSA: Vision-Language-Action Models with Plug-and-Play Safety Constraint Layer," arXiv:2512.11891, 2025.
[8] Kapoor, P. et al., "Constrained Decoding for Robotics Foundation Models," arXiv:2509.01728, 2025.
[9] Guo, J. et al., "On Robustness of Vision-Language-Action Model against Multi-Modal Perturbations," ICLR 2026, arXiv:2510.00037.
[10] Xiao, W. et al., "SafeDiffuser: Safe Planning with Diffusion Probabilistic Models," ICLR 2025, arXiv:2306.00148.
[11] Ma, H. et al., "CoDiG: Constraint-Aware Diffusion Guidance for Robotics: Real-Time Obstacle Avoidance," CoRL 2025, arXiv:2505.13131.
[12] Tölle, M. et al., "Towards Safe Robot Foundation Models Using Inductive Biases," arXiv:2505.10219, 2025.
[13] Agia, C. et al., "Unpacking Failure Modes of Generative Policies: Runtime Monitoring of Consistency and Progress," CoRL 2024, arXiv:2410.04640.
[14] Vovk, V. et al., "Algorithmic Learning in a Random World," Springer, 2005.
[15] Page, E.S., "Continuous Inspection Schemes," Biometrika 41(1/2):100–115, 1954.
[16] Gu, Q. et al., "SAFE: Multitask Failure Detection for Vision-Language-Action Models," NeurIPS 2025, arXiv:2506.09937.
[17] Römer, R. et al., "Failure Prediction at Runtime for Generative Robot Policies," NeurIPS 2025, arXiv:2510.09459.
[18] Feldman, A.O. et al., "Conformal Safety Monitoring for Flight Testing: A Case Study in Data-Driven Safety Learning," arXiv:2511.20811, 2025.
[19] Gibbs, I. and Candès, E., "Adaptive Conformal Inference Under Distribution Shift," NeurIPS 2021, arXiv:2106.00170.
[20] Kim, J. et al., "Modular Safety Guardrails Are Necessary for Foundation-Model-Enabled Robots in the Real World," arXiv:2602.04056, 2026.
[21] Chi, C. et al., "Diffusion Policy: Visuomotor Policy Learning via Action Diffusion," RSS 2023, arXiv:2303.04137.
[22] Lee, S. et al., "Behavior Generation with Latent Actions," ICML 2024, arXiv:2403.03181.
[23] Zhao, T.Z. et al., "Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware," RSS 2023, arXiv:2304.13705.