WHERE IS FAILURE PREDICTABLE? STAGE-WISE ATTRIBUTION IN VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language-Action (VLA) models map observations and instructions to robot actions, but they still fail in ways that can damage objects or leave the robot in unrecoverable states. A VLA processes each step in three phases: it aligns the scene with the instruction, reasons over the aligned representation, and decodes an action. A failure can originate in any of these phases, yet existing failure detectors return a single score for the whole policy. We derive a theoretical characterization of uncertainty and failure predictability across the VLA processing pipeline. In particular, we establish an exact decomposition of predictive action entropy into alignment, reasoning, and decoding terms with nonnegative population values, and prove that failure predictability likewise decomposes into nonnegative gains from phase-restricted scores. Motivated by our theoretical results, we develop STAGE, a computational framework for phase-wise failure prediction. STAGE extracts alignment and reasoning signals from their stage representations, computes decoding uncertainty directly from the decoder, and integrates the three signals to predict failure in a single VLA forward pass. Extensive experiments demonstrate the effectiveness of STAGE across three VLA families (OpenVLA, π_0, and π_0-FAST), four LIBERO suites, and SimplerEnv. STAGE achieves the highest AUROC in most settings, outperforming the strongest baseline by up to 10.6 points, and outperforms all evaluated baselines on real-robot Agilex rollouts. Stage-targeted perturbations further validate the phase-wise attribution: each score responds to its corresponding stage while remaining unaffected by perturbations to later stages, revealing where failure signals arise in the VLA pipeline.

1 INTRODUCTION

Vision-Language-Action (VLA) models are a compelling paradigm for generalist robotic manipulation, unifying perception, language, and control end-to-end (Ma et al., 2026). Built on vision-language model (VLM) priors and fine-tuned on robot demonstrations, VLAs such as the RT series (Brohan et al., 2023; Zitkovich et al., 2023), OpenVLA (Kim et al., 2024), and π_0 (Black et al., 2024; Intelligence et al., 2025) map raw observations and instructions directly to low-level actions. However, recent studies show that VLA policies remain brittle under modest shifts in viewpoint, initial state, or scene layout, where failures can lead to collisions, dropped objects, or irrecoverable states (Fei et al., 2025). Understanding and predicting such failures requires looking beyond the final action to how it is produced. A VLA processes each step through three phases: it aligns the visual scene with the instruction, reasons over the aligned representation, and decodes the result into an action. Failures can arise at any of these phases (Soh & Lim, 2026; Zhou et al., 2024). For example, in Figure 1, the grasp is suboptimal as the scene is misaligned with the instruction (alignment), the gripper then moves to the wrong position due to flawed spatial reasoning (reasoning), and these errors propagate into the decoded actions until the task fails (decoding). Each phase can therefore expose a failure signal before it becomes evident in

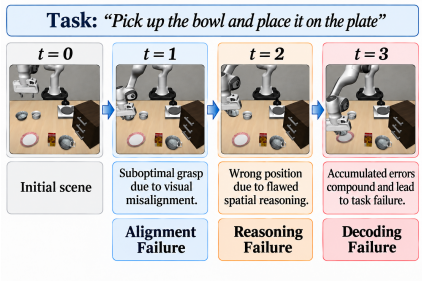


Figure 1: An OpenVLA failure rollout on LIBERO. Stage labels illustrate possible explanations rather than verified architectural causes.

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.

the final action, motivating failure prediction over the entire VLA processing pipeline rather than its output alone.

Existing failure detectors for VLAs score the policy as a whole. Token-uncertainty scores read the probability or entropy of the decoded action tokens (Smith, 2014; Karli et al., 2025), and calibration studies ask how well such confidences track success (Zollo & Zemel, 2025; Valle et al., 2025). Embedding-distance detectors flag rollouts whose internal features lie far from those of successful training rollouts (Chen et al., 2026; Xu et al., 2025). Learned detectors train a failure classifier on the policy's internal representations (Gu et al., 2025), possibly over time (Tang et al., 2026), and some couple detection with intervention (Yang et al., 2026; Qi et al., 2026). Each returns one score for the whole policy, so it cannot say in which phase the failure signal arises, and the decoder's uncertainty alone is a weak signal: a sharply concentrated distribution can still favor an incorrect action, and in our experiments token-level scores remain weak. In this paper, we study the following two fundamental research questions. **RQ1:** What theoretical principles govern how failure-relevant uncertainty is structured across the alignment, reasoning, and decoding phases of a VLA? **RQ2:** How can we develop a computational framework that leverages these phase-wise signals to predict rollout failure and attribute the prediction to individual phases?

To address these questions, we first develop a theoretical characterization of how uncertainty and failure predictability are structured across the VLA pipeline. We formulate alignment, reasoning, and decoding as successive phases and establish an exact decomposition of predictive action entropy into phase-specific terms with population values. Building on this decomposition, we further characterize failure predictability through phase-restricted scores and show that the predictive gain can be decomposed into contributions from individual phases. Together, these results provide a principled foundation for understanding where failure-relevant signals arise across the VLA pipeline.

Building on this characterization, we introduce STAGE (Stage-wise Tracing of Action-Generation Errors), a computational framework for phase-wise failure prediction and attribution. STAGE extracts alignment and reasoning signals from their stage-specific representations, computes decoding uncertainty directly from the decoder, and integrates the three signals to predict failure in a single forward pass of the frozen VLA. Extensive evaluations across three VLA families (OpenVLA, π_0, and π_0-FAST), four LIBERO suites, and SimplerEnv show that STAGE achieves the highest AUROC in most settings, outperforming the strongest baseline by up to 10.6 points. Moreover, on real-robot Agilex rollouts, STAGE further outperforms all evaluated baselines. Stage-targeted perturbations validate the phase-wise attribution, showing that each phase score responds to its corresponding stage while remaining unaffected by perturbations to later stages.

2 PRELIMINARIES

In this section, we introduce the VLA processing pipeline and formally define the problem of failure prediction and phase-wise attribution.

Notations. We use lowercase bold letters ($\mathbf{x}$) for vectors, uppercase bold letters ($\mathbf{X}$) for matrices, and calligraphic letters ($\mathcal{A}$) for sets. At timestep t, let $X^t = \{X_V^t, X_{\text{text}}^t\}$ denote the multimodal input, consisting of the visual observation X_V^t and text instruction X_{text}^t. We use H_c to denote the action-chunk horizon and $A_{0:H_c-1}^t$ to denote the action chunk generated at timestep t. For an autoregressive decoder, we further define the action prefix as $A_{<h}^t = (A_0^t, \ldots, A_{h-1}^t)$, with $A_{<0}^t = \varnothing$. Table 6 in Appendix A summarizes the key symbols.

VLA processing pipeline. A VLA maps multimodal observations and instructions to robot actions through a sequence of internal processing stages. We represent a typical VLA pipeline at timestep t:

$$X^t \xrightarrow{f_\theta} Z^t \xrightarrow{g_\phi} f_{0:L-1}^t \xrightarrow{k_\psi} y_{0:H_c-1}^t \xrightarrow{d_\omega} A_{0:H_c-1}^t. \tag{1}$$

Here, f_θ encodes the multimodal input into the aligned representation Z^t; g_ϕ processes this representation through a reasoning stack with hidden states $f_{0:L-1}^t = (f_0^t, \ldots, f_{L-1}^t)$; k_ψ maps the resulting representation to the action-relevant decoder interface $y_{0:H_c-1}^t$; and d_ω produces the action chunk $A_{0:H_c-1}^t$. Accordingly, we view the VLA computation through three phases: *alignment*, represented by Z^t; *reasoning*, represented by $f_{0:L-1}^t$; and *decoding*, which maps the resulting representation to actions. This phase-wise view provides the basis for the theoretical analysis in Section 3.

For discrete decoders, A_h^t denotes an action token and H_c the token-sequence length. For continuous decoders, A_h^t instead denotes a physical action vector and H_c its chunk horizon. We distinguish deterministic hidden states, decoder-interface representations, sampled action tokens, and physical actions throughout the paper. In particular, token entropy corresponds to physical-action entropy only when the decoding map is one-to-one; compressed tokenizations require an explicit mapping. For autoregressive policies, the action distribution at horizon h is conditioned on $A_{<h}^t$, whereas flow-matching or diffusion policies may generate the entire action chunk jointly.

The observation X^t may only partially reveal the latent environment state S^t, unavailable to the failure detector. Our theoretical analysis therefore characterizes the information available from the observable representations rather than assuming access to the underlying environment state.

Problem definition. Given a frozen VLA, the observations and internal representations available up to timestep t, and the currently generated action chunk $A_{0:H_c-1}^t$, our goal is to predict whether the current episode will eventually fail and identify where the corresponding failure signal arises within the VLA processing pipeline. Formally, the detector produces a failure score $\hat{y}_e^t$ together with three phase-wise scores associated with alignment, reasoning, and decoding. These scores should support both accurate failure prediction and attribution of the predictive signal to individual phases.

3 Theoretical Analysis

In this section, we develop a theoretical analysis of how uncertainty and failure predictability are structured across the alignment, reasoning, and decoding phases of a VLA. First, we derive an exact decomposition of predictive action entropy into three phase-specific terms and characterize their nonnegative population values. Second, we establish a decomposition of failure predictability into nonnegative gains from phase-restricted scores, providing a theoretical basis for attributing failure signals to individual phases.

3.1 Three-phase decomposition of the action distribution

Consider decision step t and action token h of the pipeline in Equation (1). Let $C_h^t = (X_{\text{text}}^t, t, A_{<h}^t)$ collect the instruction, the elapsed time, and the action prefix; it excludes the image. Let F_h^t be the final hidden state of $f_{0:L-1}^t$ at the position that emits token h, taken after the final normalization layer. The frozen policy computes

$$Z^t = f_\theta(X^t), \qquad f_{0:L-1}^t = g_\phi(Z^t, A_{<h}^t), \qquad \pi_h^t(\cdot) = \mathrm{softmax}\big(W_\psi F_h^t\big), \tag{2}$$

where W_ψ is the linear output head and the softmax is renormalized over the action vocabulary $\mathcal{A}$. All maps are fixed; randomness comes from the rollout distribution of $(X^t, A_{<h}^t)$. Let $\bar{A}_h^t \sim \pi_h^t$ be drawn given F_h^t, independently of everything else. It coincides with the executed token under temperature-one sampling. Every quantity below depends only on π_h^t, so greedy decoding is also covered. Expectations are over episodes, a timestep with weight $1/T_e$, and the token index. Write $W_{0,h}^t = C_h^t$, $W_{1,h}^t = (C_h^t, Z^t)$ and $W_{2,h}^t = (C_h^t, Z^t, F_h^t)$, and suppress (t, h) when unambiguous.

All proofs for this section are in Appendix C.

Definition 3.1 (Nested probe families)**.** For $k \in \{0, 1, 2\}$, let $\mathcal{V}_k$ be the log-linear family $q(a \mid W_k) \propto \exp\big(\beta^\top \varphi_k(W_k, a)\big)$ with nested feature maps $\varphi_0 \subseteq \varphi_1 \subseteq \varphi_2$ that read, respectively, fixed embeddings of C, of (C, Z), and of (C, Z, F), where φ_2 includes the features $e_a \otimes F$ with e_a the one-hot encoding of a. The *predictive action entropy* given W_k is

$$H_{\mathcal{V}_k}(\bar{A} \mid W_k) = \inf_{q \in \mathcal{V}_k} \mathbb{E}\big[\,\mathrm{CE}(\pi, q(\cdot \mid W_k))\big], \qquad \mathrm{CE}(\pi, q) = -\textstyle\sum_a \pi(a) \log q(a), \tag{3}$$

with minimizers $q_0^\star, q_1^\star$ for $k = 0, 1$.

Assumption 3.1 (Decoder realizability)**.** The policy's action distribution π_h^t of (2) lies in the richest probe family $\mathcal{V}_2$ of Definition 3.1: there is a parameter β_2 such that $q_{\beta_2}(\cdot \mid W_{2,h}^t) = \pi_h^t$ for every step t and token h. We write $\pi \in \mathcal{V}_2$.

Realizability holds for linear-head decoders such as OpenVLA, since $\log \pi(a)$ is linear in $e_a \otimes F$ up to normalization. The minimizers $q_0^\star, q_1^\star$ exist because π has full support: the soft-target cross-entropy is convex and diverges along every parameter direction that changes q.

For probes $q_0 \in \mathcal{V}_0$ and $q_1 \in \mathcal{V}_1$, define the phase terms of token h at step t as

$$s^t_{1,h} = \sum_a \pi^t_h(a) \log \frac{q_1(a \mid W^t_{1,h})}{q_0(a \mid W^t_{0,h})}, \qquad s^t_{2,h} = \mathrm{KL}\left(\pi^t_h \| q_1(\cdot \mid W^t_{1,h})\right), \qquad s^t_{3,h} = H(\pi^t_h). \quad (4)$$

Theorem 3.1 (Three-phase decomposition)**.** *For every step t and token h,*

$$\mathrm{CE}\left(\pi^t_h, q_0(\cdot \mid W^t_{0,h})\right) = \underbrace{s^t_{1,h}}_{\textit{Phase 1: alignment}} + \underbrace{s^t_{2,h}}_{\textit{Phase 2: reasoning}} + \underbrace{s^t_{3,h}}_{\textit{Phase 3: decoding}}, \quad (5)$$

with $s^t_{2,h}, s^t_{3,h} \geq 0$. Summing over the chunk,

$$\sum_{h=0}^{H_c-1} \mathrm{CE}\left(\pi^t_h, q_0(\cdot \mid W^t_{0,h})\right) = \sum_{h=0}^{H_c-1} \big[\textit{Phase 1} + \textit{Phase 2} + \textit{Phase 3}\big]. \quad (6)$$

Under Assumption 3.1, at the optimal probes $q_k = q^\star_k$,

$$\mathbb{E}[s_{1,h}] = \mathbb{E}\big[\mathrm{KL}(q^\star_1 \| q^\star_0)\big], \qquad \mathbb{E}[s_{2,h}] = \mathbb{E}\big[\mathrm{KL}(\pi_h \| q^\star_1)\big], \qquad \mathbb{E}[s_{3,h}] = \mathbb{E}\big[H(\pi_h)\big], \quad (7)$$

all nonnegative. Moreover, $\mathbb{E}[s_{1,h}] = H_{\mathcal{V}_0} - H_{\mathcal{V}_1}$, $\mathbb{E}[s_{2,h}] = H_{\mathcal{V}_1} - H_{\mathcal{V}_2}$, and $\mathbb{E}[s_{3,h}] = H_{\mathcal{V}_2}$, so the three phases sum to the predictive action entropy $H_{\mathcal{V}_0}(\bar{A}_h \mid C_h)$.

The decomposition in Theorem 3.1 separates predictive action uncertainty according to where information enters the VLA pipeline: alignment and reasoning are the drops in predictive action entropy as the probe gains access to Z and then to F, and decoding is the entropy that remains. While the decomposition holds pointwise for arbitrary probes, nonnegativity of the alignment and reasoning contributions is a population property attained by the optimal nested probes. The decoding term is given by the policy entropy, whereas the alignment and reasoning terms depend on the declared probe and quantify the additional predictive information accessible from their corresponding stages.

Proposition 3.1 (When a phase vanishes)**.** *Under Assumption 3.1, at the optimal probes: (i) $\mathbb{E}[s_2] = 0$ if and only if $\pi \in \mathcal{V}_1$ almost surely; (ii) $\mathbb{E}[s_1] = 0$ if and only if $q^\star_1 = q^\star_0$ almost surely; (iii) $\mathbb{E}[s_3] > 0$, since a softmax has full support.*

The reasoning phase is therefore zero only for a policy that is exactly softmax-linear in the probe inputs. For example, let C be constant, $Z = (B_1, B_2)$ with independent fair bits, $F = B_1 \oplus B_2$, and $\pi(1 \mid F) = \sigma(\beta(2F - 1))$ with $\beta > 0$. No linear probe on (B_1, B_2) improves on the constant $1/2$, so the population phases are $(0,\ \log 2 - h_2(\sigma(\beta)),\ h_2(\sigma(\beta)))$ with h_2 the binary entropy in nats (verification in Appendix C).

3.2 Characterizing Phase-Wise Failure Predictability

Let $Y \in \{0, 1\}$ be the eventual failure of the episode, assigned to each of its steps. Each phase has an input that is the representation of its own stage: $U^t_1 = Z^t$, the aligned representation (the projected visual tokens together with the prompt-token embeddings that enter the reasoning stack with them); $U^t_2 = f^t_{L-1}$, the final layer of the reasoning stack over the context, whose last position is F^t_0; and $U^t_3 = (s^t_{3,h})_h$, the exact Phase-3 terms. A *phase-restricted score* is a nonnegative function of one phase input, aggregated over the chunk as in (6):

$$S^t_k = \sum_{h=0}^{H_c-1} \sigma_k(U^t_k, h), \qquad k \in \{1, 2\}, \qquad S^t_3 = \sum_{h=0}^{H_c-1} s^t_{3,h}, \quad (8)$$

where $\sigma_k(u, h) = \mathrm{softplus}\left(\phi_k(g_k(u), h)\right)$ composes a learned reader g_k of the stage representation with a horizon-conditioned map ϕ_k. The softplus keeps $S_1, S_2 \geq 0$, like the population values of Phases 1 and 2 in (7), and its closure contains the zero function ($\phi_k \to -\infty$).

Definition 3.2 (Phase-subset predictors)**.** For $P \subseteq \{1, 2, 3\}$, let $\mathcal{G}_P$ be the family of logistic failure predictors that read only the chunk-level scores of the phases in P,

$$q(Y = 1 \mid S^t_P) = \sigma\Big(b + \textstyle\sum_{k \in P} w_k S^t_k\Big),$$

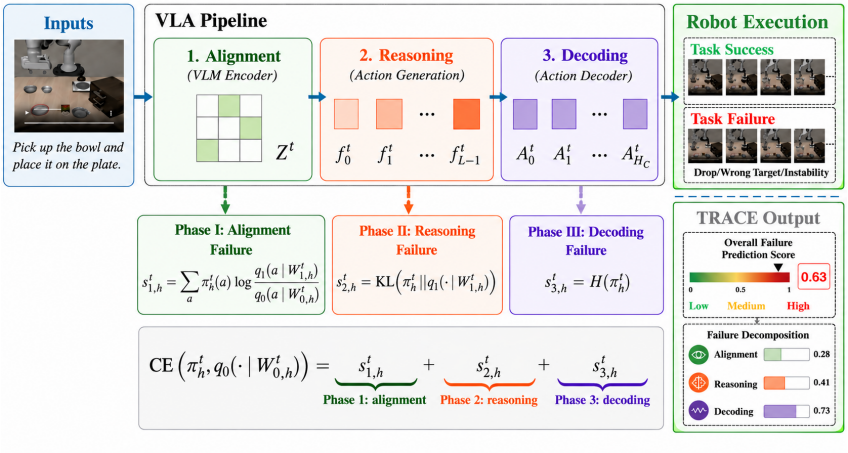


Figure 2: Overview of the STAGE framework. Each phase contributes one score read only from its own stage: **Phase I** (Alignment) from the aligned representation, **Phase II** (Reasoning) from the reasoning stack, and **Phase III** (Decoding) from the decoder output. The scores are aggregated over the action chunk and fed into a linear head that predicts the failure probability.

with $w_k, b \in \mathbb{R}$ and σ_1, σ_2 ranging over their declared families. Let

$$L(P) = \inf_{q \in \mathcal{G}_P} \mathbb{E}\big[-\log q(Y \mid S_P^t)\big],$$

where, as above, the expectation weights each timestep of an episode by $1/T_e$. The families are nested: for $P \subseteq P'$, every $q \in \mathcal{G}_P$ is attained in $\mathcal{G}_{P'}$ by setting $w_k = 0$ for $k \in P' \setminus P$.

Proposition 3.2 (Phase decomposition of failure predictability)**.** *For any ordering k_1, k_2, k_3 of the phases, with $P_j = \{k_1, \ldots, k_j\}$,*

$$L(\varnothing) - L(\{1,2,3\}) = \sum_{j=1}^{3} \underbrace{\big[L(P_{j-1}) - L(P_j)\big]}_{\text{gain of phase } k_j}, \qquad \text{each gain } \geq 0. \tag{9}$$

Proposition 3.2 provides the theoretical basis for phase-wise failure attribution. The nonnegative gains follow directly from the nesting of the phase-restricted predictor families and do not depend on the quality of the learned readers. This characterization motivates STAGE to restrict each phase score to its corresponding stage and the failure predictor to these scores alone. A phase's contribution can then be quantified by the degradation in held-out prediction when that phase is removed. The proposition guarantees nonnegative gains in population log loss; in our experiments, we additionally report the corresponding change in AUROC as an empirical measure of discriminative performance.

4 STAGE Framework

In this section, we develop STAGE as a computational framework that instantiates the theoretical characterization in Section 3 for phase-wise failure prediction and attribution. First, we construct phase-restricted failure scores for alignment, reasoning, and decoding, with each score accessing only its corresponding VLA stage. Second, we aggregate these scores over the action chunk to predict eventual failure and attribute the predictive signal to individual phases. Finally, we train the framework end-to-end using only episode-level failure outcomes, without phase-level supervision. STAGE requires only a single forward pass of the frozen VLA and is illustrated in Figure 2.

4.1 Framework Overview

The theoretical results in Section 3 motivate a phase-wise computational framework for failure prediction. Theorem 3.1 characterizes where failure-relevant information can be read across the VLA pipeline: alignment from Z^t, reasoning from f_{L-1}^t, and decoding from the action distribution π_h^t. Building on this characterization, STAGE learns phase-restricted failure scores for alignment and reasoning, while computing the decoding score directly from the policy output when available. Proposition 3.2 further motivates restricting the failure predictor to these phase scores, so that the predictive contribution of each phase can be quantified by removing its score.

As illustrated in Figure 2, STAGE consists of three main components. First, *stage readers* extract compact representations from the alignment output Z^t and reasoning output f_{L-1}^t of the frozen VLA. Second, *phase-wise scoring* converts these representations into horizon-specific alignment and reasoning scores, while the decoding score is derived from the decoder output; the resulting scores are aggregated over the action chunk. Third, *failure prediction and attribution* uses only the aggregated phase scores to predict eventual episode failure and measures each phase's contribution by removing its score from the predictor. Together, these components operationalize the theoretical phase structure using a single forward pass of the frozen VLA.

4.2 Phase-Wise Failure Scoring

Here, we aim to extract a failure-predictive signal from each VLA phase while preserving the phase restrictions established in Section 3. The key challenge is that, unlike the decoding phase, the alignment and reasoning contributions are not directly available in closed form from a forward pass. To address this, we learn *phase-restricted failure scores*: rather than estimating the theoretical phase values, each score accesses only the representation of its corresponding stage and learns the failure-relevant signal available from that stage. Accordingly, for each action token h of chunk t, STAGE produces three nonnegative scores: an alignment score $S_{1,h}^t$, a reasoning score $S_{2,h}^t$, and a decoding score $S_{3,h}^t$. The first two are learned from their stage representations, while the third is computed directly from the decoder when available.

Alignment and reasoning score. For the two learned phases, STAGE uses the phase inputs $U_1^t = Z^t$ and $U_2^t = f_{L-1}^t$ defined in Section 3. A stage-specific reader first condenses each representation into a compact code. The alignment reader $u_1^t = g_1(Z^t)$ summarizes the visual and prompt tokens together with their cross-modal interaction, while the reasoning reader $u_2^t = g_2(f_{L-1}^t)$ summarizes the final reasoning layer together with its last position F_0^t, which emits the first action token. Because each reader accesses only its corresponding stage, the phase restriction in Section 3 holds by construction. A horizon-conditioned map ϕ_k then converts the stage code into a phase score,

$$S_{k,h}^t = \mathrm{softplus}\big(\phi_k(u_k^t, h)\big), \qquad k \in \{1, 2\}.$$

This corresponds to $\sigma_k(U_k^t, h)$ in (8). The softplus enforces nonnegativity, consistent with the population phase values in Section 3, while the horizon embedding allows the shared stage representation u_k^t to produce distinct scores across the action horizon.

Decoding score. In contrast to the learned alignment and reasoning scores, the decoding score is directly available from the policy output for discrete-action decoders. Specifically, STAGE uses the exact Phase-3 term, $S_{3,h}^t = H(\pi_h^t)$, requiring no additional learning. For continuous-action decoders such as π_0, where no per-token softmax distribution is available, we instead derive $S_{3,h}^t$ from statistics of the predicted actions at horizon h, while retaining the same alignment and reasoning scores. We detail this extension in Appendix D.

4.3 Chunk-Level Failure Prediction and Attribution

The three phase scores capture complementary failure signals, but combining them with raw VLA features or a highly coupled prediction module would blur their phase-specific information and undermine phase-wise attribution. Proposition 3.2 suggests a simple principle: preserve the phase restriction through the final prediction by allowing the failure predictor to access only the three phase scores. We therefore aggregate each phase independently and combine the resulting chunk-level scores with a lightweight prediction head. This design integrates information across phases while maintaining a direct correspondence between each predictor input and VLA phase.

Prediction head. We aggregate the horizon-level scores into chunk-level phase scores, $S_k^t = \sum_{h=0}^{H_c-1} S_{k,h}^t$ for $k \in \{1, 2, 3\}$, following (6) and (8). A logistic head then predicts eventual episode failure using only these three scores,

$$\hat{y}^t = \sigma\big(b + w_1 S_1^t + w_2 S_2^t + w_3 S_3^t\big), \tag{10}$$

which instantiates the predictor family $\mathcal{G}_{\{1,2,3\}}$ in Definition 3.2. The predictor receives no raw VLA representations, per-token score statistics, or information from previous chunks. This restriction

preserves the phase-specific structure required by Proposition 3.2 and enables the contribution of each phase to be evaluated independently.

Phase-wise attribution. The phase-restricted prediction head enables a direct measure of each phase's contribution. Specifically, we remove phase k and retrain the same predictor and objective using only $(S_j^t)_{j\neq k}$, yielding a predictor in $\mathcal{G}_{\{1,2,3\}\setminus\{k\}}$. Following Proposition 3.2, the resulting degradation in predictive performance quantifies the additional failure signal contributed by phase k. In experiments, we measure this degradation by the drop in held-out AUROC (Section 5.3).

4.4 Training objective.

STAGE is trained using only episode-level outcomes, without phase-level supervision. Each chunk t of episode e inherits the eventual failure label $Y_e \in \{0, 1\}$, and the phase scores are learned jointly through their contribution to failure prediction. Following the population objective $L(\{1, 2, 3\})$ in Definition 3.2, we minimize the episode-balanced binary cross-entropy

$$\mathcal{L} = -\frac{1}{|\mathcal{E}|} \sum_{e\in\mathcal{E}} \frac{1}{T_e} \sum_{t=1}^{T_e} \left[Y_e \log \hat{y}_e^t + (1 - Y_e) \log \left(1 - \hat{y}_e^t\right) \right], \tag{11}$$

where T_e is the number of chunks in episode e. The factor $1/T_e$ gives each episode equal weight regardless of its length. We jointly optimize the stage readers g_1, g_2, the horizon-conditioned maps ϕ_1, ϕ_2, and the prediction head $(\mathbf{w}, b)$, while keeping the underlying VLA frozen.

5 Experiments

We evaluate STAGE along four dimensions: (i) overall performance across six simulation benchmarks and one real-robot setup, where STAGE exceeds most baselines (Section 5.2). (ii) ablation studies to quantify the contribution of individual components and assess the impact of phases (Section 5.3). (iii) scalability tests to demonstrate that STAGE scales efficiently (Section 5.4). (iv) real-world datasets to demonstrate the effectiveness in reality, a perturbation test to show the phase sensitivity, and a case study to show how the phase scores indicate where in the pipeline the failure signal arises (Section 5.5).

5.1 Settings

Benchmarks. We evaluate on six simulation benchmarks, LIBERO-10, -Spatial, -Object, and -Goal from LIBERO (Liu et al., 2023) and two suites from SimplerEnv (Li et al., 2024b) (Google Robot and WidowX, pooled into one row in Tables 1 and 2), and one real-robot Agilex setup. **Backbones.** We monitor OpenVLA (Kim et al., 2024) and π_0-FAST, which decode discrete action tokens, and π_0 (Black et al., 2024), which decodes continuous action chunks by flow matching. **Baselines.** We compare against three families: *token uncertainty* (Huang et al., 2023) (max/avg token probability and entropy; discrete decoders only, reported in Appendix F); *embedding distance*: Mahalanobis (Chen et al., 2026), Euclidean and cosine k-NN (Zhang, 2022), RND (He et al., 2024), and LogpZO (Xu et al., 2025); and *supervised*: SAFE-LSTM and SAFE-MLP (Gu et al., 2025) and INSIGHT (Karli et al., 2025), which detects per-step failure from token-uncertainty features. **Metrics.** We report test AUROC averaged over random splits, with training/validation/test splits following SAFE (Gu et al., 2025). Dataset details are in Appendix E (Tables 7 and 8), the search grid in Appendix J (Table 14), and implementation and baseline details in Appendix K.

5.2 Effectiveness Analysis

Tables 1 to 3 report AUROC with the OpenVLA, π_0, and π_0-FAST backbones. Token-uncertainty baselines are reported in Table 10 (Appendix F). STAGE achieves the highest AUROC in most of the settings. Appendix G splits every cell into seen-task and unseen-task AUROC (Tables 11 to 13). **OpenVLA.** STAGE is best on all five rows of Table 1. The largest gain over the strongest baseline (SAFE-MLP) is on LIBERO-Goal (82.58 vs. 74.05, +8.5 pts); on the other benchmarks the margin is 0.5–3.0 pts. Token-level metrics (Table 10) alone is a weak failure signal, and the learned alignment and reasoning phases drive detection. π_0**.** It is best on

LIBERO-Spatial by 10.6 pts over LogpZO (64.37 vs. 53.74, with every baseline below 54), on LIBERO-10 by 2.9 pts over Mahalanobis Distance, and on SimplerEnv by 5.6 pts over Cosine k-NN. It is second on LIBERO-Goal (85.28 vs. 87.54 for SAFE-MLP). On LIBERO-Object the embedding-distance methods are stronger (80.72 vs. 87.79 for Cosine k-NN), suggesting that π_0's features are already well separated for distance-based detection on this suite. **π_0-FAST.** STAGE is best on LIBERO-Spatial, LIBERO-Object, and LIBERO-Goal, by up to 5.0 pts on LIBERO-Object, and third on LIBERO-10, 0.3 pts behind Cosine k-NN and within seed variance.

Table 1: AUROC↑ (%) on LIBERO and SimplerEnv under OpenVLA. **Bold**=best, underline=second best per row. SR is the success rate (%) of the backbone policy over all rollouts of the benchmark.

		Embedding distance					Supervised			Ours
Dataset	SR	Mahal.	Eucl. k-NN	Cos. k-NN	RND	LogpZO	SAFE-LSTM	SAFE-MLP	INSIGHT	**STAGE**
LIBERO-10	55.0	58.61 ±1.71	62.93 ±0.97	62.70 ±0.73	51.78 ±3.61	51.36 ±1.45	65.41 ±1.41	67.01 ±2.56	55.11 ±0.54	**68.73** ±1.11
LIBERO-spatial	79.6	70.97 ±3.59	74.14 ±0.83	77.11 ±0.52	63.53 ±0.52	66.71 ±1.58	81.72 ±7.34	84.38 ±1.23	64.28 ±0.43	**87.35** ±1.16
LIBERO-object	70.4	61.20 ±7.70	67.38 ±4.73	71.26 ±1.32	55.83 ±3.09	64.73 ±2.17	73.87 ±7.38	81.99 ±0.79	55.32 ±0.59	**82.48** ±1.98
LIBERO-goal	77.6	63.74 ±1.78	63.29 ±1.80	68.89 ±4.23	55.63 ±3.69	63.62 ±2.31	70.14 ±4.56	74.05 ±4.06	60.78 ±1.59	**82.58** ±1.66
SimplerEnv	8.1	72.37 ±1.92	80.78 ±4.80	82.13 ±1.37	80.58 ±1.05	36.02 ±4.18	80.99 ±3.10	82.57 ±5.90	64.84 ±0.06	**84.08** ±4.61

Table 2: AUROC↑ (%) on LIBERO and SimplerEnv with the π_0 backbone. INSIGHT is computed from the spread over the action statistics as a proxy (‡). **Bold**=best, underline=second best per row. SR is the success rate (%) of the backbone policy over all rollouts of the benchmark.

		Embedding distance					Supervised			Ours
Dataset	SR	Mahal.	Eucl. k-NN	Cos. k-NN	RND	LogpZO	SAFE-LSTM	SAFE-MLP	INSIGHT‡	**STAGE**
LIBERO-10	61.0	78.72 ±5.73	73.88 ±6.26	73.08 ±6.10	75.80 ±6.29	77.70 ±6.84	71.84 ±8.79	77.32 ±8.69	62.63 ±3.19	**81.59** ±0.74
LIBERO-spatial	95.5	41.30 ±6.15	47.91 ±4.29	46.44 ±6.22	47.69 ±4.37	53.74 ±5.05	51.62 ±6.65	48.21 ±1.05	46.23 ±1.15	**64.37** ±1.92
LIBERO-object	88.0	72.31 ±2.49	87.24 ±3.13	**87.79** ±3.14	83.55 ±2.51	84.45 ±7.02	57.39 ±3.44	79.50 ±8.00	67.60 ±8.91	80.72 ±4.58
LIBERO-goal	83.8	83.42 ±1.90	82.04 ±0.96	81.73 ±1.50	84.91 ±1.73	83.98 ±0.98	81.60 ±1.21	**87.54** ±0.49	78.10 ±1.73	85.28 ±2.40
SimplerEnv	63.9	66.82 ±3.82	73.45 ±7.76	74.07 ±9.94	67.88 ±3.35	66.81 ±3.48	67.01 ±7.03	65.76 ±6.34	69.78 ±2.45	**79.67** ±5.88

Table 3: AUROC↑ (%) on LIBERO with the π_0-FAST backbone. **Bold**=best, underline=second best per row. SR is the success rate (%) of the backbone policy over all rollouts of the benchmark.

		Embedding distance					Supervised			Ours
Dataset	SR	Mahal.	Eucl. k-NN	Cos. k-NN	RND	LogpZO	SAFE-LSTM	SAFE-MLP	INSIGHT	**STAGE**
LIBERO-10	47.2	90.40 ±1.28	91.09 ±0.88	**93.26** ±0.77	90.62 ±2.25	92.35 ±1.91	93.23 ±1.02	71.46 ±15.75	92.75 ±0.71	92.92 ±0.79
LIBERO-spatial	84.0	86.29 ±5.81	88.46 ±1.10	87.93 ±1.54	83.04 ±6.75	89.00 ±3.67	87.36 ±2.81	75.51 ±9.84	83.45 ±2.79	**90.55** ±3.54
LIBERO-object	93.4	82.07 ±2.48	92.79 ±0.95	91.98 ±1.43	89.28 ±0.50	88.47 ±3.57	92.36 ±1.64	68.77 ±15.40	81.56 ±6.90	**97.82** ±0.22
LIBERO-goal	66.4	90.12 ±2.73	91.85 ±0.55	91.75 ±0.85	90.76 ±0.30	89.68 ±0.66	93.73 ±0.51	83.79 ±6.68	83.12 ±0.82	**94.23** ±1.31

5.3 Ablation Analysis

Table 4: Phase ablation of STAGE on OpenVLA LIBERO-spatial. **Bold**=best.

Group	Phases P	AUROC
One phase	Phase 1	73.49 ±13.54
	Phase 2	64.72 ±14.13
	Phase 3	57.87 ±7.45
Two phases	Phase 1 + Phase 2	86.43 ±1.30
	Phase 1 + Phase 3	77.16 ±9.78
	Phase 2 + Phase 3	74.52 ±12.16
Full	Phases 1 + 2 + 3	**87.35** ±1.16

Table 4 reports the contribution of each phase and of their combinations on OpenVLA LIBERO-Spatial: each row trains the phase-subset predictor of Definition 3.2 on the listed phases. **Single phases.** The alignment phase is the strongest on its own (73.49), followed by the reasoning phase (64.72); the decoding phase, the softmax entropy on which token-level baselines rely, is the weakest (57.87). All three single-phase predictors vary strongly across seeds. **Combinations.** Combining the alignment and reasoning phases gives the AUROC (86.43), well above either phase alone, so the two learned phases carry complementary failure signal. Stage-targeted perturbations of the frozen VLA (Appendix H, Figure 5) show that each phase score responds to a perturbation of its own stage which indicates the phase's sensitivity.

5.4 Scalability Analysis

Training cost of STAGE stops growing once episodes exceed its 64-step training window: as episodes become 40 times longer (10 to 400 steps), the time per epoch rises only from 8.1 s to 27.1 s, and scoring one new step takes 7–11 ms at every length. The full timing study is in Appendix I (Figure 6).

5.5 Case Study

We deploy the pretrained π_0 (Black et al., 2024) on an Agilex manipulator and collect 201 rollouts on three lifting and pick-and-place tasks (98 successes, 103 failures; Figure 4, Appendix E). STAGE reaches 98.25 AUROC (Table 5), 29.4 points above the strongest baseline, SAFE-MLP, showing that it remains effective on a real robot. Figure 3 shows a representative failure. When the robot misses its first grasp of the sponge (t=0–80), the failure probability rises with the alignment score; when it drops the object on the second attempt (t=200–240), the decoding score spikes; and during the failed regrasp, the reasoning score increases. The phase scores thus rise where the failure unfolds and indicate which phase carries the warning signal.

Table 5: AUROC (%) on Agilex data. **Bold**: best; underline: second.

Method	AUROC↑ (%)
Mahalanobis Dist.	56.49 ±5.20
Euclidean k-NN	58.56 ±3.93
Cosine k-NN	59.86 ±1.54
RND	57.50 ±4.60
LogpZO	53.59 ±5.75
SAFE-LSTM	64.03 ±10.79
SAFE-MLP	68.88 ±2.15
STAGE	**98.25** ±1.83

Phase attribution under controlled perturbations. Since each phase score reads only its own stage, perturbing one stage should change the scores of that stage and later stages while leaving earlier ones unchanged. We test this by perturbing one stage of the frozen OpenVLA on LIBERO-10. The results in Figure 5 (Appendix H) confirm this order: the change in the alignment score grows monotonically with the pixel-noise level, noise inside the LLM layers changes the reasoning score while the alignment score moves by less than 0.1%, and the change in the decoding score grows monotonically with the decoder temperature while leaving the other two unchanged.

6 Related Work

Vision-Language-Action models. VLA models unify visual perception, language understanding, and action generation within a single policy (Ma et al., 2026; Ghosh et al., 2024; Zhen et al., 2024; Li et al., 2024a), decoding either discrete action tokens (OpenVLA (Kim et al., 2024)) or flow-based continuous actions (π_0 (Black et al., 2024; Intelligence et al., 2025)). Despite strong performance, they remain brittle under distribution shift and frequently produce unsafe or hallucinated actions (Fei et al., 2025; Soh & Lim, 2026), motivating principled approaches for runtime failure detection.

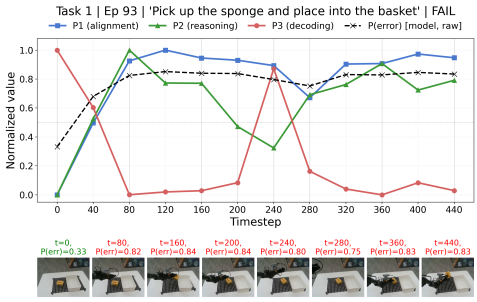


Figure 3: Examples to show how each phase failure score changes along timestep.

Failure detection and uncertainty estimation. Existing VLA failure detectors fall into five families: *output-distribution* (Chen et al., 2026; Liu et al., 2024b; Sinha et al., 2024), *embedding-based* (Gu et al., 2025; Grewal et al., 2024), *density-estimation* (Xu et al., 2025), *supervisor-augmented* (Yang et al., 2026; Liu et al., 2024b;a; Gokmen et al., 2023), and *sampling-based consistency* (Agia et al., 2024) methods. These methods use different predictive signals and supervision; scalar detection performance and architectural diagnosis require distinct evidence. The broader LLM/VLM literature studies token-level uncertainty (Huang et al., 2023; Ling et al., 2024; Xiao & Wang, 2021), semantic consistency (Kuhn et al., 2023; Chen & Mueller, 2024; Lin et al., 2024), and object hallucination in multimodal outputs (Zhou et al., 2024; Chen & Mueller, 2024; Favero et al., 2024), with recent adaptations to VLA models (Zollo & Zemel, 2025; Valle et al., 2025; Tang et al., 2026; Soh & Lim, 2026). INSIGHT (Karli et al., 2025) models temporal token-level uncertainty for help prediction under strong and weak supervision.

7 Conclusion

We studied VLA failure prediction through the alignment, reasoning, and decoding phases of the VLA pipeline. Our theory decomposes both predictive action entropy and failure predictability into per-phase terms. Guided by it, STAGE scores each phase from its own stage and predicts failure from these scores alone, using a single forward pass of the frozen VLA. Across three VLA families, LIBERO, SimplerEnv, and a real robot, STAGE attains the highest AUROC in most settings, and stage-targeted perturbations confirm that each phase score is driven by its own stage, indicating where in the pipeline the failure signal arises.

REPRODUCIBILITY STATEMENT

For the theory, Section 3 states every definition and assumption, and Appendix C gives complete proofs. Section 4 specifies the STAGE architecture and training objective, and Appendix D describes the phase inputs for continuous-action decoders. Appendix E describes the benchmarks, backbone policies, per-task success rates and seen/unseen splits (Tables 7 to 9). Appendix J gives the full hyperparameter grid and the validation-only selection rule, and Appendix K gives the remaining implementation details and the baseline protocols. All results are averaged over three fixed seeds whose task splits are shared by every method, and Appendices F and G report the per-split results behind Tables 1 to 3.

AI USE STATEMENT

Large language models were used only for language polishing and wording refinement. Specifically, LLM assistance was limited to improving grammar, clarity, conciseness, and presentation of author-written text. All scientific ideas, problem formulation, theoretical results, proofs, algorithms, experimental design, implementation, data processing, numerical results, analysis, and conclusions were produced and verified by the authors. LLMs were not used to generate experimental results, fabricate data, perform reviewer simulation for reported claims, or make autonomous scientific decisions. The authors take full responsibility for the final content of the paper.

REFERENCES

Christopher Agia, Rohan Sinha, Jingyun Yang, Ziang Cao, Rika Antonova, Marco Pavone, and Jeannette Bohg. Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*, volume 270 of *Proceedings of Machine Learning Research*, pp. 689–723. PMLR, 2024. URL https://proceedings.mlr.press/v270/agia25a.html.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael 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, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. *CoRR*, abs/2410.24164, 2024. doi: 10.48550/ARXIV.2410.24164. URL https://doi.org/10.48550/arXiv.2410.24164.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S. Ryoo, Grecia Salazar, Pannag R. Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong T. Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: robotics transformer for real-world control at scale. In Kostas E. Bekris, Kris Hauser, Sylvia L. Herbert, and Jingjin Yu (eds.), *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*, 2023. doi: 10.15607/RSS.2023.XIX.025. URL https://doi.org/10.15607/RSS.2023.XIX.025.

Jiuhai Chen and Jonas Mueller. Quantifying uncertainty in answers from any language model and enhancing their trustworthiness. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 5186–5200. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.283. URL https://doi.org/10.18653/v1/2024.acl-long.283.

Lingling Chen, Zongyao Lyu, and William J. Beksi. Reconvla: An uncertainty-guided and failure-aware vision-language-action framework for robotic control. *CoRR*, abs/2604.16677, 2026.

doi: 10.48550/ARXIV.2604.16677. URL https://doi.org/10.48550/arXiv.2604.16677.

Alessandro Favero, Luca Zancato, Matthew Trager, Siddharth Choudhary, Pramuditha Perera, Alessandro Achille, Ashwin Swaminathan, and Stefano Soatto. Multi-modal hallucination control by visual information grounding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pp. 14303–14312. IEEE, 2024. doi: 10.1109/CVPR52733.2024.01356. URL https://doi.org/10.1109/CVPR52733.2024.01356.

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. *CoRR*, abs/2510.13626, 2025. doi: 10.48550/ARXIV.2510.13626. URL https://doi.org/10.48550/arXiv.2510.13626.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R. Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In Dana Kulic, Gentiane Venture, Kostas E. Bekris, and Enrique Coronado (eds.), *Robotics: Science and Systems XX, Delft, The Netherlands, July 15-19, 2024*, 2024. doi: 10.15607/RSS.2024.XX.090. URL https://doi.org/10.15607/RSS.2024.XX.090.

Cem Gokmen, Daniel Ho, and Mohi Khansari. Asking for help: Failure prediction in behavioral cloning through value approximation. In *IEEE International Conference on Robotics and Automation, ICRA 2023, London, UK, May 29 - June 2, 2023*, pp. 5821–5828. IEEE, 2023. doi: 10.1109/ICRA48891.2023.10161004. URL https://doi.org/10.1109/ICRA48891.2023.10161004.

Yashvir S. Grewal, Edwin V. Bonilla, and Thang D. Bui. Improving uncertainty quantification in large language models via semantic embeddings. *CoRR*, abs/2410.22685, 2024. doi: 10.48550/ARXIV.2410.22685. URL https://doi.org/10.48550/arXiv.2410.22685.

Qiao Gu, Yuanliang Ju, Shengxiang Sun, Igor Gilitschenski, Haruki Nishimura, Masha Itkina, and Florian Shkurti. SAFE: multitask failure detection for vision-language-action models. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/392d0d05e2f514063e6ce6f8b370834c-Abstract-Conference.html.

Nantian He, Shaohui Li, Zhi Li, Yu Liu, and You He. Rediffuser: Reliable decision-making using a diffuser with confidence estimation. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17921–17933. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/he24e.html.

Yuheng Huang, Jiayang Song, Zhijie Wang, Huaming Chen, and Lei Ma. Look before you leap: An exploratory study of uncertainty measurement for large language models. *CoRR*, abs/2307.10236, 2023. doi: 10.48550/ARXIV.2307.10236. URL https://doi.org/10.48550/arXiv.2307.10236.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *CoRR*, abs/2504.16054, 2025.

doi: 10.48550/ARXIV.2504.16054. URL https://doi.org/10.48550/arXiv.2504.16054.

Ulas Berk Karli, Ziyao Shangguan, and Tesca Fitzgerald. INSIGHT: inference-time sequence introspection for generating help triggers in vision-language-action models. *CoRR*, abs/2510.01389, 2025. doi: 10.48550/ARXIV.2510.01389. URL https://doi.org/10.48550/arXiv.2510.01389.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Paul Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2024. URL https://proceedings.mlr.press/v270/kim25c.html.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=VD-AYtP0dve.

Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, Hang Li, and Tao Kong. Vision-language foundation models as effective robot imitators. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024a. URL https://openreview.net/forum?id=lFYj0oibGR.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, 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. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3705–3728. PMLR, 2024b. URL https://proceedings.mlr.press/v270/li25c.html.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Generating with confidence: Uncertainty quantification for black-box large language models. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=DWkJCSxKU5.

Chen Ling, Xujiang Zhao, Xuchao Zhang, Wei Cheng, Yanchi Liu, Yiyou Sun, Mika Oishi, Takao Osaki, Katsushi Matsuda, Jie Ji, Guangji Bai, Liang Zhao, and Haifeng Chen. Uncertainty quantification for in-context learning of large language models. In Kevin Duh, Helena Gómez-Adorno, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pp. 3357–3370. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.NAACL-LONG.184. URL https://doi.org/10.18653/v1/2024.naacl-long.184.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: benchmarking knowledge transfer for lifelong robot learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/8c3c666820ea055a77726d66fc7d447f-Abstract-Datasets_and_Benchmarks.html.

Huihan Liu, Shivin Dass, Roberto Martín-Martín, and Yuke Zhu. Model-based runtime monitoring with interactive imitation learning. In *IEEE International Conference on Robotics and Automation, ICRA 2024, Yokohama, Japan, May 13-17, 2024*, pp. 4154–4161. IEEE, 2024a. doi: 10.1109/ICRA57147.2024.10611038. URL https://doi.org/10.1109/ICRA57147.2024.10611038.

Huihan Liu, Yu Zhang, Vaarij Betala, Evan Zhang, James Liu, Crystal Ding, and Yuke Zhu. Multi-task interactive robot fleet learning with visual world models. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4286–4313. PMLR, 2024b. URL https://proceedings.mlr.press/v270/liu25g.html.

Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. A survey on vision-language-action models for embodied AI. *IEEE Trans. Neural Networks Learn. Syst.*, 37(7):3031–3051, 2026. doi: 10.1109/TNNLS.2025.3650584. URL https://doi.org/10.1109/TNNLS.2025.3650584.

Carl Qi, Xiaojie Wang, Silong Yong, Stephen Sheng, Huitan Mao, Sriram Srinivasan, Manikantan Nambi, Amy Zhang, and Yesh Dattatreya. Self-refining vision language model for robotic failure detection and reasoning. *CoRR*, abs/2602.12405, 2026. doi: 10.48550/ARXIV.2602.12405. URL https://doi.org/10.48550/arXiv.2602.12405.

Rohan Sinha, Amine Elhafsi, Christopher Agia, Matthew Foutter, Edward Schmerling, and Marco Pavone. Real-time anomaly detection and reactive planning with large language models. In Dana Kulic, Gentiane Venture, Kostas E. Bekris, and Enrique Coronado (eds.), *Robotics: Science and Systems XX, Delft, The Netherlands, July 15-19, 2024*, 2024. doi: 10.15607/RSS.2024.XX.114. URL https://doi.org/10.15607/RSS.2024.XX.114.

Ralph C. Smith. *Uncertainty Quantification - Theory, Implementation, and Applications*. Computational science and engineering. SIAM, 2014. ISBN 978-1-611973-21-1. URL http://bookstore.siam.org/cs12/.

Harold Soh and Eugene Lim. Action hallucination in generative visual-language-action models. *CoRR*, abs/2602.06339, 2026. doi: 10.48550/ARXIV.2602.06339. URL https://doi.org/10.48550/arXiv.2602.06339.

Yanchuan Tang, Taowen Wang, Yuefei Chen, Boxuan Zhang, Qiang Guan, and Ruixiang Tang. Shifting uncertainty to critical moments: Towards reliable uncertainty quantification for VLA model. *CoRR*, abs/2603.18342, 2026. doi: 10.48550/ARXIV.2603.18342. URL https://doi.org/10.48550/arXiv.2603.18342.

Pablo Valle, Chengjie Lu, Shaukat Ali, and Aitor Arrieta. Evaluating uncertainty and quality of visual language action-enabled robots. *CoRR*, abs/2507.17049, 2025. doi: 10.48550/ARXIV.2507.17049. URL https://doi.org/10.48550/arXiv.2507.17049.

Yijun Xiao and William Yang Wang. On hallucination and predictive uncertainty in conditional language generation. In Paola Merlo, Jörg Tiedemann, and Reut Tsarfaty (eds.), *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pp. 2734–2744. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.EACL-MAIN.236. URL https://doi.org/10.18653/v1/2021.eacl-main.236.

Chen Xu, Tony Khuong Nguyen, Emma Dixon, Christopher Rodriguez, Patrick Tree Miller, Robert Lee, Paarth Shah, Rares Ambrus, Haruki Nishimura, and Masha Itkina. Can we detect failures without failure data? uncertainty-aware runtime failure detection for imitation learning policies. *CoRR*, abs/2503.08558, 2025. doi: 10.48550/ARXIV.2503.08558. URL https://doi.org/10.48550/arXiv.2503.08558.

Yifan Yang, Zhixiang Duan, Tianshi Xie, Fuyu Cao, Pinxi Shen, Peili Song, Chenyang Zhao, Piaopiao Jin, Guokang Sun, Shaoqing Xu, Yangwei You, and Jingtai Liu. FPC-VLA: A vision-language-action framework with a supervisor for failure prediction and correction. *Expert Syst. Appl.*, 316:131742, 2026. doi: 10.1016/J.ESWA.2026.131742. URL https://doi.org/10.1016/j.eswa.2026.131742.

Shichao Zhang. Challenges in KNN classification. *IEEE Trans. Knowl. Data Eng.*, 34(10): 4663–4675, 2022. doi: 10.1109/TKDE.2021.3049250. URL https://doi.org/10.1109/TKDE.2021.3049250.

Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 61229–61245. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/zhen24a.html.

Yiyang Zhou, Chenhang Cui, Jaehong Yoon, Linjun Zhang, Zhun Deng, Chelsea Finn, Mohit Bansal, and Huaxiu Yao. Analyzing and mitigating object hallucination in large vision-language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=oZDJKTlOUe.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong T. Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Conference on Robot Learning, CoRL 2023, 6-9 November 2023, Atlanta, GA, USA*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

Thomas P. Zollo and Richard S. Zemel. Confidence calibration in vision-language-action models. *CoRR*, abs/2507.17383, 2025. doi: 10.48550/ARXIV.2507.17383. URL https://doi.org/10.48550/arXiv.2507.17383.

A NOTATIONS

Table 6 lists the key symbols used in the paper.

Table 6: Key symbols and descriptions.

Symbol	Description
X^t	Multi-modal input (image and instruction) at step t
Z^t	Aligned representation (projected visual tokens and prompt-token embeddings)
$f^t_{0:L-1}$	Reasoning-stack hidden states; f^t_{L-1} is the final layer
F^t_h	Final hidden state at the position that emits action token h
H_c, $A^t_{0:H_c-1}$	Action chunk size and action chunk
π^t_h	Decoder distribution over action bins for token h at step t
C^t_h	Instruction context: instruction, elapsed time, action prefix (no image)
$W^t_{k,h}$, $\mathcal{V}_k$	Probe inputs and nested probe families (Definition 3.1)
$s^t_{k,h}$	Phase-k term of the three-phase decomposition (Theorem 3.1)
U^t_k	Phase input of stage k: $U^t_1 = Z^t$, $U^t_2 = f^t_{L-1}$, $U^t_3 = (s^t_{3,h})_h$
g_k, ϕ_k	Learned stage reader and horizon-conditioned phase map of STAGE
$S^t_{k,h}$	Per-token phase score of STAGE; $S^t_{3,h} = H(\pi^t_h)$ for discrete decoders
S^t_k	Chunk-level phase score $\sum_h S^t_{k,h}$
$\mathcal{G}_P$, $L(P)$	Phase-subset predictors and their optimal log loss (Definition 3.2)
Y, $\hat{y}^t$	Eventual failure and predicted failure probability
$H(\cdot)$, $I(\cdot;\cdot)$, CE, KL	Entropy, mutual information, cross-entropy, KL divergence

B Shannon Mutual Information for a Frozen Backbone

This appendix explains why Section 3 measures information usable by declared probes rather than Shannon mutual information. With the notation of Section 3, the Shannon counterpart of the three-phase decomposition degenerates for a frozen deterministic backbone.

Proposition B.1 (Shannon phases collapse). $H(\bar{A} \mid C) = I(\bar{A}; Z \mid C) + I(\bar{A}; F \mid C, Z) + \mathbb{E}[H(\pi)]$, *and* $I(\bar{A}; F \mid C, Z) = 0$ *because* F *is a function of* (C, Z). *If* C *also contains the image, then* $I(\bar{A}; Z \mid C) = 0$ *as well.*

Rewriting these terms as divergences does not help, since $\mathbb{E}_Z \, \mathrm{KL}(P(\bar{A} \mid C, Z) \| P(\bar{A} \mid C)) = I(\bar{A}; Z \mid C)$. The probe-based phases of Theorem 3.1 reduce to these Shannon terms when the probe families are unrestricted: then $q_k^\star = P(\bar{A} \mid W_k)$ and each $H_{\mathcal{V}_k}$ equals $H(\bar{A} \mid W_k)$, so the reasoning phase vanishes. With restricted families it need not: in the XOR example of Section 3, the probe phases are $(0,\ \log 2 - h_2(\sigma(\beta)),\ h_2(\sigma(\beta)))$, whereas the Shannon terms assign $\log 2 - h_2(\sigma(\beta))$ to alignment and zero to reasoning, because $\bar{A}$ is a fair bit and $H(\bar{A} \mid Z) = h_2(\sigma(\beta))$.

Proof of Proposition B.1. The chain rule gives $H(\bar{A} \mid C) = I(\bar{A}; Z \mid C) + I(\bar{A}; F \mid C, Z) + H(\bar{A} \mid C, Z, F)$. Given F, $\bar{A}$ is drawn from $\pi(\cdot \mid F)$ independently of (C, Z), so $H(\bar{A} \mid C, Z, F) = \mathbb{E}[H(\pi)]$. By (2), $F = g_\phi(Z, A_{<h})$ is a function of (C, Z) because $A_{<h}$ is part of C. Conditional on (C, Z), F is therefore constant and $I(\bar{A}; F \mid C, Z) = 0$. If C contains X, the same argument applies to $Z = f_\theta(X)$. □

C Proofs for the Phase Decomposition

Proof of Theorem 3.1. **Pointwise identity.** $H(\pi) + \mathrm{KL}(\pi \| q_1) = \mathrm{CE}(\pi, q_1)$, and adding $\sum_a \pi(a) \log(q_1(a)/q_0(a))$ gives $\mathrm{CE}(\pi, q_0)$. This proves (5); nonnegativity of s_2 and s_3 is that of KL divergence and entropy. Summing over h gives (6).

For the population values, write $R_k(q) = \mathbb{E}[\mathrm{CE}(\pi, q(\cdot \mid W_k))]$, so that $H_{\mathcal{V}_k} = \inf_{q \in \mathcal{V}_k} R_k(q)$.

Phase 3. For every q, $\mathrm{CE}(\pi, q) = H(\pi) + \mathrm{KL}(\pi \| q) \geq H(\pi)$ with equality if and only if $q = \pi$ (Gibbs' inequality). Since $\pi \in \mathcal{V}_2$, the infimum is attained at π and $H_{\mathcal{V}_2} = \mathbb{E}[H(\pi)]$.

Phase 2. $H_{\mathcal{V}_1} - H_{\mathcal{V}_2} = R_1(q_1^\star) - \mathbb{E}[H(\pi)] = \mathbb{E}[\mathrm{KL}(\pi \| q_1^\star)] \geq 0$.

Phase 1. Parametrize $q_\beta(a \mid W_1) = \exp(\beta^\top \varphi_1(W_1, a) - \log \mathcal{Z}_\beta(W_1))$. The gradient of R_1 is $\mathbb{E}[\sum_a (q_\beta(a \mid W_1) - \pi(a)) \varphi_1(W_1, a)]$, and it vanishes at the minimizer β_1 because the parametrization is unconstrained. Hence, for every fixed vector γ,

$$\mathbb{E}\Big[\sum_a \pi(a)\, \gamma^\top \varphi_1(W_1, a)\Big] = \mathbb{E}\Big[\sum_a q_1^\star(a \mid W_1)\, \gamma^\top \varphi_1(W_1, a)\Big]. \tag{12}$$

Because $\varphi_0 \subseteq \varphi_1$, there is a vector β_0' with $\beta_0^\top \varphi_0(W_0, a) = \beta_0'^\top \varphi_1(W_1, a)$. Therefore

$$\log \frac{q_1^\star(a \mid W_1)}{q_0^\star(a \mid W_0)} = (\beta_1 - \beta_0')^\top \varphi_1(W_1, a) - \big[\log \mathcal{Z}_{\beta_1}(W_1) - \log \mathcal{Z}_{\beta_0}(W_0)\big].$$

The first term is linear in φ_1, so by (12) its expectation is the same under π and under $q_1^\star$. The bracket does not depend on a, and both π and $q_1^\star$ sum to one, so its expectation is also the same. Hence

$$H_{\mathcal{V}_0} - H_{\mathcal{V}_1} = \mathbb{E}\Big[\sum_a \pi(a) \log \frac{q_1^\star}{q_0^\star}\Big] = \mathbb{E}\Big[\sum_a q_1^\star(a) \log \frac{q_1^\star}{q_0^\star}\Big] = \mathbb{E}[\mathrm{KL}(q_1^\star \| q_0^\star)] \geq 0.$$

These three differences are the expectations of s_1, s_2 and s_3 at $q_k = q_k^\star$, which gives (7), and they sum to $H_{\mathcal{V}_0}$. □

Proof of Proposition 3.1. (i) If $\pi \in \mathcal{V}_1$ almost surely, Gibbs' inequality makes π the minimizer of R_1, so $q_1^\star = \pi$ and $\mathbb{E}[s_2] = 0$. Conversely, $\mathbb{E}[\mathrm{KL}(\pi \| q_1^\star)] = 0$ forces $\pi = q_1^\star \in \mathcal{V}_1$ almost surely. (ii) $\mathbb{E}[\mathrm{KL}(q_1^\star \| q_0^\star)] = 0$ if and only if $q_1^\star = q_0^\star$ almost surely. (iii) A softmax over $|\mathcal{A}| \geq 2$ actions assigns positive probability to every action, so $H(\pi) > 0$. □

Verification of the XOR example. $\mathbb{E}[\pi(1 \mid F)] = \frac{1}{2}$ because F is a fair bit, so the constant probe $q_0^\star = \frac{1}{2}$ is optimal in $\mathcal{V}_0$. For $\mathcal{V}_1$ with features $(1, B_1, B_2)$, the constant $\frac{1}{2}$ satisfies (12): $\mathbb{E}[\pi(1) - \frac{1}{2}] = 0$, and $\mathbb{E}[B_1(\pi(1) - \frac{1}{2})] = \frac{1}{4}[(\sigma(\beta) - \frac{1}{2}) + (\sigma(-\beta) - \frac{1}{2})] = 0$, and likewise for B_2. The objective is strictly convex in the logistic parameters because the features are affinely independent, so $q_1^\star = \frac{1}{2}$. Then $\mathbb{E}[s_2] = \mathbb{E}[\mathrm{KL}(\pi \| \frac{1}{2})] = \log 2 - h_2(\sigma(\beta))$, using $h_2(\sigma(\beta)) = h_2(\sigma(-\beta))$. $\mathbb{E}[s_3] = h_2(\sigma(\beta))$. □

Proof of Proposition 3.2. The sum in (9) telescopes. By nesting, for every $q \in \mathcal{G}_{P_{j-1}}$ and $\epsilon > 0$, the family $\mathcal{G}_{P_j}$ contains a predictor with expected log loss at most that of q plus ϵ. Taking infima gives $L(P_j) \leq L(P_{j-1})$. □

D Phase Inputs for Continuous-Action Decoders

Flow-matching decoders such as π_0 expose no discrete per-step distribution, so the Phase-3 entropy has no closed form. The alignment and reasoning stages, however, exist in every VLA regardless of how actions are decoded. For π_0 and open-pi-zero, STAGE therefore reads Phases 1 and 2 from the same stage representations as for discrete decoders, with the same readers, phase maps and head: $U_1^t = Z^t$, the projected SigLIP image tokens (base and wrist cameras on LIBERO) together with the prompt-token embeddings, and $U_2^t = f_{L-1}^t$, the final layer of the PaliGemma prefix over the context ($d = 2048$). Only Phase 3 differs. For continuous-action decoders such as π_0, $S_{3,h}^t$ is instead derived from statistics of the predicted actions at horizon h, as in Section 4; the same choice is used for the real-robot Agilex experiments.

E Datasets and Benchmarks

We evaluate across three benchmark families chosen to span complementary axes: task diversity (LIBERO), decoder paradigm and embodiment coverage (SimplerEnv), and physical-robot evaluation (Agilex). Together they cover both discrete-token and flow-matching policies and both seen and zero-shot task splits.

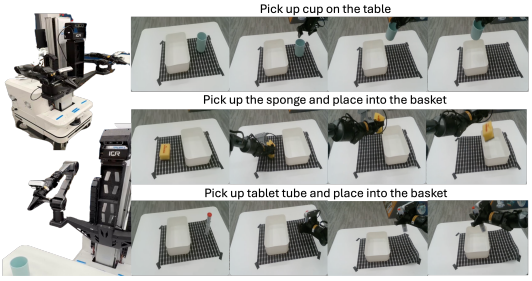


Figure 4: Illustration of real-world experiment settings and example rollouts collected.

LIBERO. LIBERO (Liu et al., 2023) is a lifelong-manipulation benchmark of four task suites, each isolating a different generalization axis: **LIBERO-Spatial** varies object *layout* while fixing the object set, stressing spatial grounding; **LIBERO-Object** fixes the layout and varies the *objects*, stressing object identification; **LIBERO-Goal** fixes objects and layout and varies the *instruction/goal*, stressing language conditioning; and **LIBERO-10** consists of ten long-horizon, multi-stage tasks combining all three axes, and is the most error-prone suite. Each suite has 10 tasks with 50 demonstrations per task. We use OpenVLA, π_0, and π_0-FAST rollouts; LIBERO-10 is the suite reported in SAFE (Gu et al., 2025), while the other three are additional evaluations.

SimplerEnv. SimplerEnv (Li et al., 2024b) provides simulation environments tuned to reproduce the visual and dynamical statistics of real robot data, enabling reproducible policy evaluation. We use two embodiments with distinct action spaces and skill profiles: **Google Robot** (4 tasks: object reaching, drawer opening/closing, and a long-horizon place-in-drawer task, replicating RT-series setups) and **WidowX** (4 tasks: pick-and-place and stacking, replicating BridgeData V2). Crucially, we collect rollouts from *two* policy families on the same tasks which are the autoregressive OpenVLA (discrete action tokens) and Open-Pi-Zero, the open-source implementation of π_0 Black et al. (2024) that we run on SimplerEnv (denoted π_0^* where the implementation matters). So that SimplerEnv tests

performance with different decoder paradigms. The two policies differ sharply in competence (Table 7): π_0^* succeeds on the majority of episodes while OpenVLA often fails, yielding very different base failure rates.

Agilex (real robot). To evaluate prediction on physical-robot data, we deploy a π_0 flow-matching policy on a physical Agilex bimanual manipulator and collect 201 rollouts over three pick-and-lift tasks (left-arm sponge grasp, single-arm cup grasp, right-arm tube grasp), comprising 98 successes and 103 failures, which is a near-balanced split that provides similar numbers of successes and failures (Figure 4; per-task success rates in Table 8). Because only three tasks are available, all are treated as seen, and we report AUROC on the held-out test rollouts of these tasks. This is the only benchmark with physical-hardware noise (sensor latency, calibration drift, contact dynamics).

Table 7: Tasks used in the SimplerEnv, with per-task success rates (%) for the two policies.

Embodiment	ID	Environment Name	SR % (π_0^*)	SR % (OpenVLA)
Google Robot	1	`google_robot_move_near_v0`	86.0	6.0
	2	`google_robot_open_drawer`	46.0	0.0
	3	`google_robot_close_drawer`	76.0	42.0
	4	`google_robot_place_apple_in_closed_top_drawer`	38.0	0.0
WidowX	1	`widowx_carrot_on_plate`	55.0	3.0
	2	`widowx_put_eggplant_in_basket`	85.0	4.0
	3	`widowx_spoon_on_towel`	81.0	0.0
	4	`widowx_stack_cube`	44.0	10.0

Table 8: Tasks used in the real-world Agilex experiments. SR is the success rate (%) of the π_0 policy over the recorded episodes of each task.

Task	Instruction	# Episodes	SR (%)
1	Pick up sponge (left arm)	51	80.4
2	Pick up cup on the table	100	35.0
3	Pick up tube (right arm)	50	44.0
All		201	48.8

Table 9: Benchmark split settings. Each seed (42, 43, 44) draws a different seen/unseen task partition; the rollout counts are the same for every seed. Following the SAFE protocol, every method selects its configuration on this validation set, and test is never used for selection. The seen and unseen AUROC of Appendix G are computed on test-seen and test-unseen, and the AUROC of Tables 1 to 3 on their union. On Agilex, all three tasks are seen.

	Rollouts			
Benchmark	Train	Validation	Test	**Total**
LIBERO, each suite (OpenVLA, π_0, π_0-FAST)	140	210	150	500
π_0^* SimplerEnv, Google Robot	100	100	200	400
π_0^* SimplerEnv, WidowX	100	100	200	400
OpenVLA SimplerEnv, Google Robot	100	100	200	400
OpenVLA SimplerEnv, WidowX	100	100	200	400
π_0 Agilex (real)	110	31	60	201

F Token-Uncertainty Baselines

Token-uncertainty scores need the decoder's per-token softmax, so they exist only for the discrete-token backbones, OpenVLA and π_0-FAST. Table 10 reports them on the same runs and splits as Tables 1 and 3, with STAGE for reference.

Table 10: AUROC↑ (%) of the token-uncertainty baselines with the OpenVLA and π_0-FAST backbones, with STAGE from Tables 1 and 3 for reference. **Bold**=best, underline=second best per row.

		Token uncertainty				Ours
Backbone	Dataset	Max Prob.	Avg Prob.	Max Ent.	Avg Ent.	**STAGE**
OpenVLA	LIBERO-10	47.67 ±0.28	46.68 ±0.51	48.76 ±0.02	47.66 ±0.70	**68.73** ±1.11
	LIBERO-spatial	60.35 ±0.58	61.39 ±0.79	60.17 ±1.44	65.45 ±0.54	**87.35** ±1.16
	LIBERO-object	52.72 ±0.86	50.77 ±0.45	55.23 ±1.24	51.67 ±0.54	**82.48** ±1.98
	LIBERO-goal	53.46 ±0.38	58.49 ±0.86	54.53 ±0.57	59.32 ±0.44	**82.58** ±1.66
	SimplerEnv	56.93 ±3.09	51.61 ±2.03	60.33 ±1.05	54.77 ±1.79	**84.08** ±4.61
π_0-FAST	LIBERO-10	70.83 ±0.81	70.61 ±0.35	60.26 ±0.62	66.26 ±1.01	**92.92** ±0.79
	LIBERO-spatial	63.68 ±0.60	56.05 ±1.58	55.78 ±0.65	58.37 ±0.68	**90.55** ±3.54
	LIBERO-object	50.25 ±2.39	70.80 ±1.39	55.34 ±0.27	64.78 ±2.18	**97.82** ±0.22
	LIBERO-goal	53.82 ±0.75	63.38 ±0.35	57.98 ±2.44	58.79 ±0.49	**94.23** ±1.31

G SEEN- AND UNSEEN-TASK RESULTS

Tables 1 to 3 report AUROC on the pooled test set. Tables 11 to 13 split each of these cells into its two parts (Table 9): test-seen/test-unseen. **Bold**=best, underline=second best per row.

Table 11: Seen- and unseen-task AUROC↑ (%) with the OpenVLA backbone, for the methods and runs of Table 1.

		Token uncertainty				Embedding distance					Supervised			Ours
Dataset	Split	Max Prob.	Avg Prob.	Max Ent.	Avg Ent.	Mahal.	Eucl. k-NN	Cos. k-NN	RND	LogpZO	SAFE-LSTM	SAFE-MLP	INSIGHT	**STAGE**
LIBERO-10	Seen	50.59 ±3.29	49.88 ±3.12	49.97 ±2.56	51.74 ±3.88	61.96 ±1.83	64.97 ±2.42	65.65 ±1.76	55.87 ±2.99	53.93 ±1.66	69.28 ±0.58	70.24 ±2.26	55.00 ±3.33	**71.10** ±1.18
	Unseen	43.54 ±4.93	42.59 ±5.34	47.22 ±3.82	42.47 ±7.11	51.45 ±3.47	59.96 ±1.92	57.49 ±1.74	44.83 ±6.92	46.21 ±1.61	59.11 ±1.63	62.59 ±5.29	54.27 ±6.04	**64.28** ±2.00
LIBERO-spatial	Seen	57.90 ±1.74	60.61 ±1.40	59.43 ±2.50	66.17 ±1.46	71.98 ±0.47	80.11 ±0.66	84.11 ±0.68	69.59 ±2.59	71.76 ±2.90	83.55 ±8.11	87.69 ±1.54	66.80 ±0.73	**90.01** ±1.56
	Unseen	66.89 ±2.61	62.33 ±3.39	62.36 ±4.77	63.63 ±2.77	64.17 ±8.24	55.26 ±2.73	61.81 ±0.26	50.97 ±6.57	60.30 ±8.21	79.73 ±4.69	84.09 ±2.40	58.67 ±1.46	**85.85** ±2.27
LIBERO-object	Seen	52.68 ±1.55	53.28 ±2.85	54.88 ±1.33	54.82 ±3.34	68.64 ±2.25	73.95 ±2.05	75.41 ±1.77	62.67 ±2.51	72.53 ±5.34	77.76 ±4.60	83.72 ±2.30	57.41 ±2.33	**87.22** ±1.91
	Unseen	53.27 ±3.99	48.10 ±5.96	55.28 ±4.25	48.19 ±7.20	50.82 ±5.73	62.78 ±20.95	72.68 ±5.04	42.64 ±0.86	54.91 ±0.54	69.10 ±12.92	**80.95** ±6.04	49.80 ±2.22	77.09 ±4.33
LIBERO-goal	Seen	50.82 ±0.83	55.83 ±3.38	51.50 ±2.19	57.02 ±2.88	69.40 ±1.24	71.93 ±1.15	73.39 ±0.86	59.92 ±1.09	61.22 ±5.92	74.67 ±4.91	78.22 ±1.57	58.09 ±2.81	**84.48** ±0.52
	Unseen	56.63 ±1.13	62.49 ±6.13	58.49 ±4.22	62.76 ±5.02	62.08 ±3.80	47.31 ±4.45	62.69 ±11.03	48.44 ±8.21	66.60 ±4.10	65.02 ±7.62	66.89 ±12.93	65.38 ±5.38	**82.62** ±4.31
SimplerEnv	Seen	61.46 ±7.04	53.32 ±3.13	78.45 ±4.78	57.86 ±5.40	96.61 ±1.74	96.65 ±1.14	97.03 ±0.46	97.21 ±0.73	40.49 ±12.04	98.96 ±0.35	98.13 ±0.48	77.54 ±2.28	**99.67** ±0.20
	Unseen	56.65 ±6.95	54.41 ±4.85	44.39 ±2.12	54.88 ±5.18	26.69 ±5.91	58.46 ±4.31	**64.64** ±0.80	55.46 ±9.91	30.16 ±7.29	49.95 ±4.63	60.29 ±3.42	47.88 ±7.65	59.77 ±1.07

Table 12: Seen- and unseen-task AUROC↑ (%) with the π_0 backbone, for the methods and runs of Table 2.

		Embedding distance					Supervised			Ours
Dataset	Split	Mahal.	Eucl. k-NN	Cos. k-NN	RND	LogpZO	SAFE-LSTM	SAFE-MLP	INSIGHT‡	**STAGE**
LIBERO-10	Seen	**91.11** ±0.33	88.48 ±0.48	88.26 ±0.52	89.30 ±0.58	89.70 ±0.75	82.08 ±5.02	89.26 ±0.51	67.82 ±5.16	**91.11** ±0.94
	Unseen	47.44 ±21.89	39.96 ±15.87	41.68 ±16.96	42.51 ±17.82	47.47 ±23.55	53.38 ±19.28	50.65 ±22.68	50.61 ±4.78	**58.46** ±11.33
LIBERO-spatial	Seen	34.04 ±14.73	51.11 ±10.46	48.33 ±11.99	48.42 ±9.75	51.53 ±6.92	62.07 ±5.10	50.54 ±4.45	42.35 ±6.54	**65.22** ±2.46
	Unseen	51.51 ±4.58	48.11 ±8.55	48.19 ±5.41	50.39 ±5.46	59.52 ±2.64	39.63 ±12.36	46.62 ±8.72	50.01 ±4.87	**66.72** ±3.38
LIBERO-object	Seen	79.70 ±2.36	89.40 ±0.64	**89.81** ±0.85	85.55 ±2.26	86.26 ±3.07	63.64 ±4.41	85.35 ±4.79	72.12 ±8.98	85.35 ±0.82
	Unseen	66.68 ±8.51	83.85 ±7.44	**84.66** ±7.54	81.04 ±7.93	82.73 ±11.52	49.99 ±0.85	73.52 ±16.62	63.05 ±9.20	75.22 ±12.69
LIBERO-goal	Seen	87.34 ±1.98	85.89 ±1.39	85.60 ±1.45	88.66 ±2.36	88.79 ±1.71	84.34 ±1.38	89.32 ±2.70	80.16 ±2.09	**90.63** ±1.75
	Unseen	69.04 ±7.06	74.14 ±2.46	74.28 ±3.08	75.34 ±3.38	74.88 ±4.12	73.97 ±3.14	**82.60** ±7.20	67.60 ±4.73	78.45 ±4.28
SimplerEnv	Seen	86.33 ±2.04	85.53 ±1.41	85.02 ±1.86	80.35 ±2.38	**87.75** ±3.04	82.37 ±3.39	78.58 ±4.01	66.70 ±3.91	86.23 ±2.40
	Unseen	53.35 ±0.43	66.10 ±10.72	69.30 ±13.96	59.61 ±3.76	57.94 ±0.99	59.86 ±8.84	58.83 ±5.01	71.28 ±4.85	**75.22** ±12.05

Table 13: Seen- and unseen-task AUROC↑ (%) with the π_0-FAST backbone, for the methods and runs of Table 3.

		Token uncertainty				Embedding distance					Supervised			Ours
Dataset	Split	Max Prob.	Avg Prob.	Max Ent.	Avg Ent.	Mahal.	Eucl. k-NN	Cos. k-NN	RND	LogpZO	SAFE-LSTM	SAFE-MLP	INSIGHT	**STAGE**
LIBERO-10	Seen	72.72 ±1.82	68.42 ±4.17	63.17 ±4.15	67.96 ±2.36	**96.82** ±1.07	94.52 ±1.19	95.95 ±1.00	94.02 ±0.76	95.76 ±0.37	96.37 ±0.93	73.75 ±15.25	94.88 ±0.75	96.50 ±0.18
	Unseen	67.39 ±4.11	59.84 ±22.08	57.43 ±6.43	51.74 ±15.34	74.09 ±10.22	78.80 ±11.78	82.83 ±10.10	77.27 ±17.65	83.22 ±10.84	80.95 ±11.37	77.30 ±16.01	81.18 ±14.10	**84.61** ±11.03
LIBERO-spatial	Seen	64.99 ±4.71	55.34 ±4.44	57.40 ±5.13	57.11 ±6.24	89.47 ±3.51	92.32 ±1.39	91.71 ±1.70	90.56 ±3.58	**92.36** ±1.60	89.77 ±5.62	79.46 ±5.97	85.39 ±5.88	91.77 ±1.98
	Unseen	57.67 ±9.47	55.83 ±8.07	51.15 ±13.89	53.70 ±10.73	80.26 ±1.98	79.53 ±1.75	**80.59** ±2.03	58.81 ±3.57	80.29 ±7.49	74.28 ±6.75	63.90 ±24.33	77.33 ±4.56	77.23 ±20.50
LIBERO-object	Seen	54.25 ±5.10	65.26 ±2.69	57.74 ±2.61	58.65 ±3.09	88.54 ±1.01	92.17 ±2.11	92.38 ±3.24	93.72 ±5.63	94.65 ±1.14	94.91 ±2.66	68.36 ±19.62	85.73 ±6.21	**99.34** ±0.46
	Unseen	48.99 ±6.86	75.69 ±0.94	57.58 ±2.20	70.33 ±3.43	79.42 ±5.58	93.28 ±1.47	92.05 ±1.30	83.49 ±8.67	85.00 ±6.29	90.45 ±5.53	72.07 ±13.17	79.31 ±8.97	**96.41** ±1.29
LIBERO-goal	Seen	50.60 ±1.06	64.69 ±2.33	56.97 ±2.68	60.13 ±4.25	94.04 ±0.68	93.47 ±0.92	93.68 ±1.05	93.08 ±1.11	95.01 ±1.05	95.12 ±0.98	87.87 ±1.68	88.93 ±2.06	**95.81** ±0.87
	Unseen	65.74 ±4.97	64.54 ±2.26	67.88 ±4.08	58.04 ±6.26	83.32 ±10.22	84.63 ±6.08	84.39 ±5.11	83.92 ±4.85	82.25 ±6.99	86.66 ±6.07	83.39 ±8.16	76.61 ±14.52	**92.36** ±2.75

H STAGE-TARGETED PERTURBATIONS

To test whether each phase score responds to its own stage, we perturb one stage of the frozen OpenVLA at a time and measure how the three phase scores of the trained STAGE change. For each seed we take 20 held-out unseen-task test episodes of LIBERO-10 (half of them failures), replay them exactly in the simulator with the logged actions, and at 20 evenly spaced steps per episode run the VLA forward pass once without and once with the perturbation. Every pass is scored by the STAGE model selected for that seed in Table 1. **Vision**: independent Gaussian noise $\mathcal{N}(0, (255\sigma)^2)$ on every pixel channel of the policy image, $\sigma \in \{0.02, 0.05, 0.10, 0.20\}$. **Reasoning**: Gaussian noise scaled by each layer's per-token activation standard deviation, added to the output of every LLM layer except the last, $\sigma \in \{0.1, 0.25, 0.5\}$; Z^t is computed before the LLM and is untouched. **Decoder**: the forward pass is unchanged and the action logits are divided by a temperature $T \in \{1.5, 2, 3\}$ before the softmax entropy, so Z^t and f^t_{L-1} are untouched.

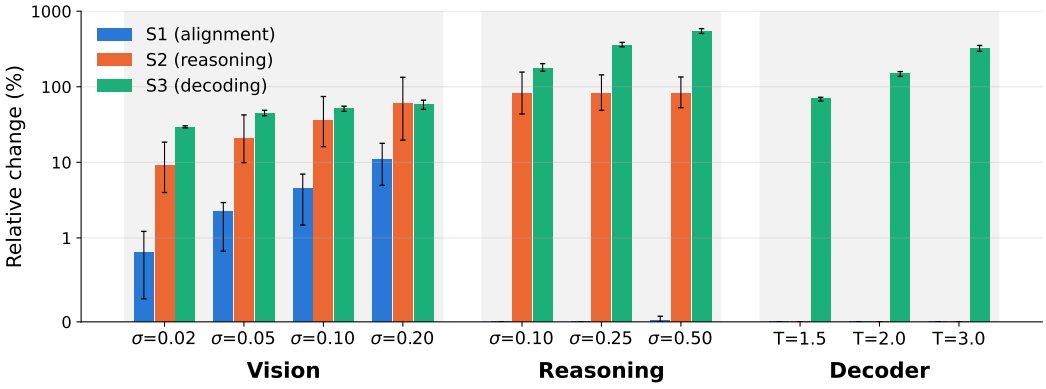


Figure 5: Relative change (%, symmetric-log scale) of the phase scores of STAGE when one VLA stage is perturbed at a time (OpenVLA, LIBERO-10, 20 unseen-task test episodes $\times$ 20 steps per seed). Each perturbation changes the score of its own stage and of later stages, and leaves the scores of earlier stages unchanged.

Figure 5 reports the relative change of each chunk-level phase score, $\sum_i |S_k^{\text{pert}} - S_k^{\text{clean}}| / \sum_i S_k^{\text{clean}}$ over the perturbed steps i. A perturbation of one stage propagates through every later stage of the pipeline, so it can change the scores of that stage and of the stages after it, but not of the stages before it. The results follow this order. Each perturbation changes the score of its own stage: the change in the alignment score grows monotonically with the pixel-noise level (0.8% to 11%), the reasoning score changes by about 80% under layer noise, and the change in the decoding score grows monotonically with the temperature (70% to 325%). Scores of earlier stages are unchanged: the alignment score changes by less than 0.1% under reasoning noise, the residual of numerical nondeterminism between forward passes, and both learned scores are exactly unchanged under temperature scaling. The later scores also respond, as expected: pixel noise moves the reasoning and decoding scores, and layer noise moves the decoding score.

I SCALABILITY ANALYSIS

Figure 6 reports the time per epoch of STAGE as the mean episode length grows from 10 to 400 steps. The episodes are simulated but use the full OpenVLA input sizes (256×4096 aligned visual tokens, the last-layer prefill context, and a 7-token action chunk), and the timed code is the STAGE training and scoring pipeline itself. Training cost stops growing once episodes exceed the 64-step training window: each epoch samples two windows per episode, so the training pass stays at about 20 s from 100 to 400 steps. Only full-episode scoring grows with length, and it does so linearly (0.8 s at 10 steps to 6.6 s at 400 steps). The total time per epoch therefore rises from 8.1 s to 27.1 s while episodes become 40 times longer. Scoring one new step takes 7–11 ms at every length, because STAGE scores each step from that step's features and needs no pass over the history. These measurements were taken on a shared NVIDIA H200 running other jobs; they characterize training and scoring cost and do not measure VLA feature extraction or the effect on control frequency.

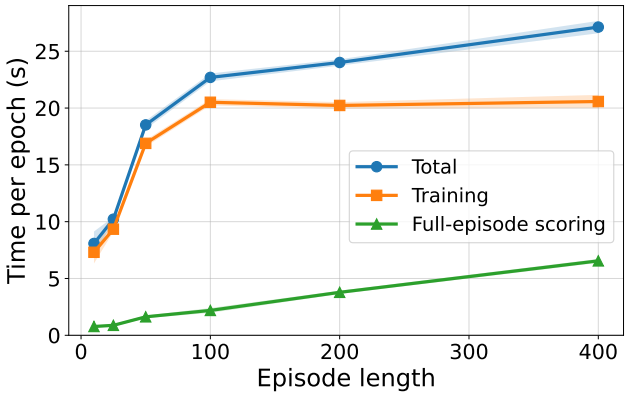


Figure 6: Time per epoch of STAGE versus mean episode length on simulated episodes with OpenVLA-sized inputs (500 episodes; mean $\pm$ std over 3 seeds $\times$ 4 epochs). Training samples two 64-step windows per episode; full-episode scoring runs over every validation step.

J Hyperparameter Search Ranges

For each benchmark and seed, STAGE selects the configuration and the training epoch with the best AUROC on the validation rollouts of the seen tasks (Table 9), as every baseline does. The grid is given in Table 14; the remaining sizes are fixed.

Table 14: Grid searched on validation for STAGE (all backbones) and the fixed sizes.

Hyperparameter	**Values**
Learning rate	$\{3 \times 10^{-4}, 10^{-3}\}$
Weight decay	$\{10^{-3}, 10^{-2}\}$
Reader output size d_{cond}	$\{8, 16\}$
Reader token MLP (fixed)	$D \to 256 \to 64$
Phase map ϕ_k (fixed)	hidden $128 \to 64$, horizon embedding 32
Dropout (fixed)	0.3

K Implementation Details

The evaluations include OpenVLA (Kim et al., 2024), π_0-style policies, and π_0-FAST. Each stage reader applies a token-wise MLP (layer normalization, $D \to 256 \to 64$) and pools the token codes as in Section 4, followed by a small MLP to d_{cond}. The phase maps ϕ_1, ϕ_2 are MLPs with two hidden layers (128 and 64 units) and a learned horizon embedding, with a softplus output. Phase 3 is the exact softmax entropy of each action token for discrete decoders and requires no learned parameters; for continuous-action decoders it is derived from statistics of the predicted actions (Appendix D). The failure head is linear in the chunk-level phase masses (Equation (10)). The network is trained with the per-chunk log loss of Equation (11) using AdamW. Each epoch samples two random windows of 64 consecutive steps per training episode, with 8 episodes per batch; training runs for up to 30 epochs with early stopping (patience 6) on the validation AUROC. All experiments are run on a single NVIDIA H200 141GB GPU. Results are averaged over three random seeds. Standard deviations are population standard deviations over seeds.