Adaptive Safety Probing for Resource-Efficient Vision-Language-Action Models

Seongbin Park [1] **Fan Zhang** [1] **Hossein Khalili** [1] **Nader Sehatbakhsh** [1]

Abstract

Foundation models (FM) are rapidly moving from cloud-centered language and vision applications into embodied robotic systems, where they must support real-time decision making under strict constraints. Vision-Language-Action (VLA) models are a prominent class of such embodied FMs and are increasingly being used as the main controller of robotic systems. While this integration enables general-purpose robotic behavior, it also introduces safety and security risks. A compromised instruction or unsafe generated action can propagate directly to the robot's actuators. Existing defenses often rely on auxiliary perception models, repeated inference, or always-on safety filters, which are poorly matched to resource-constrained robotic platforms.

We propose an adaptive inference-time safety framework for VLA models that reuses signals already computed inside the frozen policy. First, we show that intermediate hidden states encode task-relevant 3D geometric intent: a lightweight probe trained offline on successful rollouts predicts the intended grasp target and can be used as a low-cost failure monitor. Second, we reconstruct targeted action-to-vision attention maps from the VLA backbone to identify image regions driving the predicted action, avoiding the need for a separate heavyweight vision-language model. Third, we introduce Post-processing Trigger-based Purification, a check-then-defend mechanism that invokes more expensive safety interventions only when lightweight monitors detect a policy violation, target mismatch, or uncertain execution. Together, these components provide a resource-aware adaptive safety and security layer for emerging robotic foundation-model systems. Our simulation results show up to 40% runtime improvement compared to a non-adaptive safe FM.

[1]Department of Electrical and Computer Engineering, University of California, Los Angeles, USA. Secure Systems and Architectures Lab. Correspondence to: Seongbin Park <parkseongbin@ucla.edu>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

1. Introduction

Foundation models (FMs) are increasingly being deployed beyond text and image understanding, entering embodied settings where model outputs directly influence physical actions (Bommasani et al., 2021; Driess et al., 2023). Vision-Language-Action (VLA) models are one such class of embodied foundation models (Intelligence et al., 2025; Kim et al., 2024). A VLA policy consumes visual observations, language instructions, and robot state, and produces low-level or chunked robot actions (Kim et al., 2024). This makes VLAs attractive for general-purpose robotic manipulation, but it also changes the risk profile of foundation-model deployment: unsafe or adversarial model behavior can affect the physical world (Hu et al., 2025).

Safety and security mechanisms for foundation models are often implemented as external modules, such as auxiliary vision-language models (Brohan et al., 2023), repeated model queries (Wang et al., 2022), post-hoc verifiers (Cobbe et al., 2021), or always-on filters (Ouyang et al., 2022). While such defenses may be practical in cloud settings, they are expensive for emerging robotic systems operating under real-time constraints. Mobile manipulators, household robots, drones, and embedded industrial systems often have limited onboard compute, strict latency budgets (Jiang et al., 2026), and safety-critical actuation loops. In these settings, a safety mechanism must be both effective and lightweight.

The key challenge is that FMs are already facing severe latency and resource problems, and adding computationally-heavy safety or security filters will further exacerbate this issue. We propose a different design principle: An Adaptive Safe FM, where instead of adding large external safety models around a VLA, we extract safety-relevant signals from the VLA's own inference computation. Our central hypothesis is that a VLA must internally represent task-relevant geometric and semantic intent in order to act. If the robot is about to grasp an object, the policy should encode where the object is, which visual patches are relevant, and what action transition is intended. These internal signals can be used to monitor failures, detect inconsistencies, and trigger

safety interventions only when needed.

We organize our framework around three lightweight modules. First, we use intermediate hidden states as a geometric failure monitor by probing for the intended 3D grasp target. Second, we reconstruct targeted action-to-vision attention maps to localize the visual evidence used by the policy. Third, we use trigger-based action purification to enforce safety and security constraints only when a lightweight monitor detects risk. This yields an adaptive inference-time safety stack for resource-constrained VLA deployment. An overview of our approach is shown in Figure 1.

Contributions. We make the following contributions:

1. We formulate adaptive inference-time safety monitoring for VLA foundation models, where lightweight probes inspect the policy's latent computation and route only risky cases to more expensive safety interventions.

2. We demonstrate that intermediate VLA hidden states encode 3D task geometry. A lightweight hidden-state probe predicts grasp targets with high accuracy and can serve as a real-time failure signal.

3. We reconstruct action-to-vision attention maps from the VLA backbone despite Flash Attention, enabling target-object localization without a separate perception model.

4. We introduce Post-processing Trigger-based Purification (PTP), a check-then-defend mechanism that maps outputs into structured state-action transitions, checks them against safety policies, and invokes purification only when needed.

2. Problem Setup

We consider a frozen VLA policy

$$\pi_\theta(a_{t:t+H} \mid I_t, q_t, \dot{q}_t, x_{\text{lang}}), \tag{1}$$

where I_t is the visual observation, q_t and $\dot{q}_t$ are proprioceptive states, x_{lang} is the language instruction, and $a_{t:t+H} \in \mathbb{R}^{H \times d_a}$ is the predicted action chunk.

Our goal is not to retrain the VLA, but to attach a lightweight adaptive safety monitor

$$m_\phi(z_\theta) \to \{\text{safe}, \text{unsafe}, \text{uncertain}\}, \tag{2}$$

where z_θ denotes internal VLA signals computed during the standard forward pass, such as hidden states, action-token representations, and attention projections.

The final control decision is routed as

$$\tilde{a}_{t:t+H} = \begin{cases} a_{t:t+H}, & \text{if } m_\phi(z_\theta) = \text{safe}, \\ \text{Purify}(a_{t:t+H}), & \text{otherwise.} \end{cases} \tag{3}$$

$\text{Purify}(a_{t:t+H})$ can be realized in various ways depending on the context, limitations, and available capabilities. As we discuss later, we examine several possible strategies: redo, re-plan, mask, refuse, and fallback. In this adaptive strategy, costly safety interventions are triggered only when lightweight checks at inference time signal potential risk.

3. Method

Our framework consists of two key probing modules: hidden-state failure probing and attention-based target localization. These two modules are then used for our adaptive trigger-based action purification module to reduce the overall latency and resource requirements to ensure safety and security for FMs used for robotic control.

3.1. Hidden-State 3D Target Probe for Failure Detection

We first ask whether VLA hidden states encode the geometric intent needed for manipulation. While our framework is model-agnostic and can be applied to a variety of VLA architectures, in this work we focus on **OpenVLA** (Kim et al., 2024) as a representative backbone due to its widespread adoption and strong performance in recent robotic manipulation benchmarks. Importantly, our approach does not rely on any architecture-specific assumptions, and we expect the observed properties of latent geometric encoding to generalize to other VLA models that jointly reason over vision, language, and action.

An overview of our approach is shown in Figure 2. During a standard VLA forward pass, we extract the hidden state from an intermediate transformer layer ℓ:

$$h_\ell \in \mathbb{R}^{N \times D}, \tag{4}$$

where N is the token sequence length and D is the hidden dimension. In our implementation, we use layer $\ell = 8$, which we have found empirically to provide a strong balance between semantic abstraction and spatial fidelity.

We spatially pool the hidden states:

$$\bar{h}_\ell = \frac{1}{N} \sum_{i=1}^{N} h_{\ell,i}, \tag{5}$$

project the pooled representation onto a low-dimensional PCA basis,

$$z_\ell = \text{PCA}_k(\bar{h}_\ell), \tag{6}$$

and train a small MLP probe to predict the 3D grasp target:

$$\hat{p}_t = f_\phi(z_\ell), \qquad \hat{p}_t \in \mathbb{R}^3. \tag{7}$$

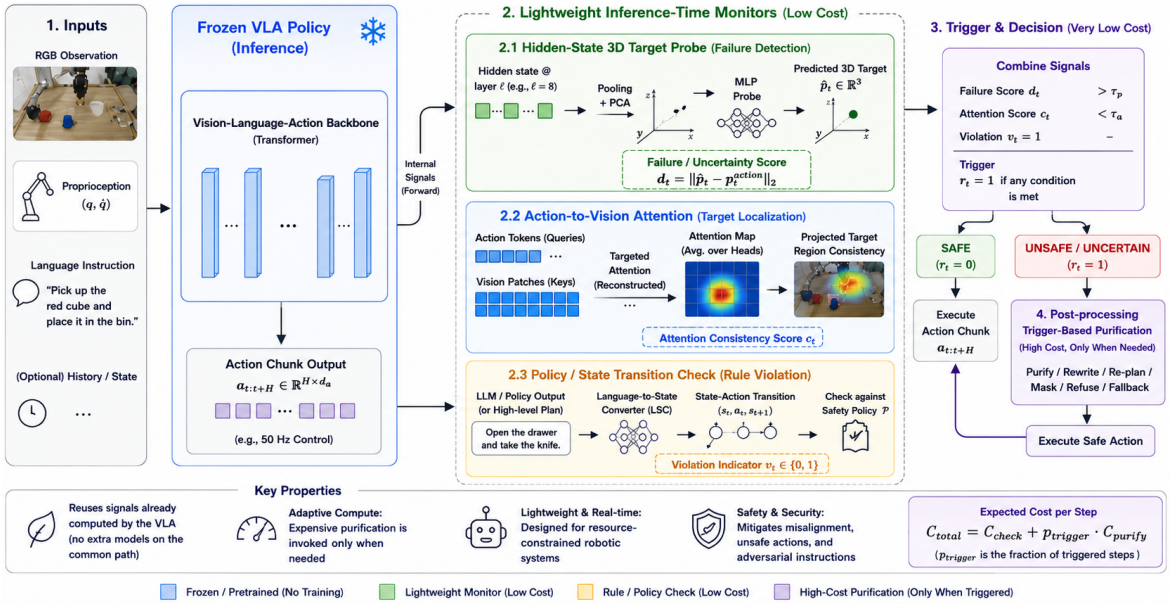

Figure 1. Overview of the proposed adaptive inference-time safety framework for resource-constrained VLA foundation models.

The probe is trained offline on successful rollouts using the mean-squared error loss

$$\mathcal{L}_{\text{probe}} = \|\hat{p}_t - p_t^\star\|_2^2, \tag{8}$$

where $p_t^\star$ is the ground-truth 3D grasp target.

At test time, the predicted target $\hat{p}_t$ provides a lightweight estimate of the VLA's intended geometric goal. This can be used for failure detection by comparing the probed target to the action-implied endpoint p_t^a:

$$d_t = \|\hat{p}_t - p_t^a\|_2 . \tag{9}$$

A failure or uncertainty trigger is raised when

$$d_t > \tau_p, \tag{10}$$

where τ_p is a task-dependent safety threshold. Intuitively, if the latent target inferred from the VLA's hidden state disagrees with the motion implied by the action chunk, the policy may be mislocalized, uncertain, or about to execute an unsafe action.

Probe performance. Trained offline on successful rollouts, the hidden-state probe achieves high 3D target prediction accuracy on LIBERO (Liu et al., 2023) benchmarks.

Results are shown in Table 1. They suggest that VLA hidden states contain precise task-relevant geometric information. Importantly, the probe reuses activations already computed during inference, avoiding a separate perception module. While demonstrated on OpenVLA, this finding points to a broader property of VLA foundation models: intermediate representations encode actionable geometric intent that can be extracted with lightweight probes for real-time safety monitoring.

Table 1. 3D grasp target prediction from OpenVLA hidden states.

Benchmark	$R^2 \uparrow$	RMSE (m) $\downarrow$
LIBERO-10	0.9900	0.0152
LIBERO-Object	0.9811	0.0128

3.2. Action-to-Vision Attention for Target Localization

The hidden-state probe estimates where the VLA intends to act in 3D. We next extract a complementary signal: which image regions the policy attends to when producing action tokens. Our hypothesis is that a VLA must attend to the task-relevant object in order to act on it. Therefore, action-to-vision attention can provide a lightweight target-localization signal for safety monitoring.

Modern VLA backbones commonly use Flash Attention (Dao et al., 2022), which improves memory and speed by avoiding explicit materialization of the full attention matrix. To preserve efficiency, we do not reconstruct the full matrix. Instead, we manually compute only the targeted submatrix between predicted action-token queries and visual-patch keys.

Let $r \in \mathbb{R}^{B\times L\times D}$ denote the residual hidden state captured at a selected transformer layer using a forward hook. We extract the hidden states corresponding to action tokens and

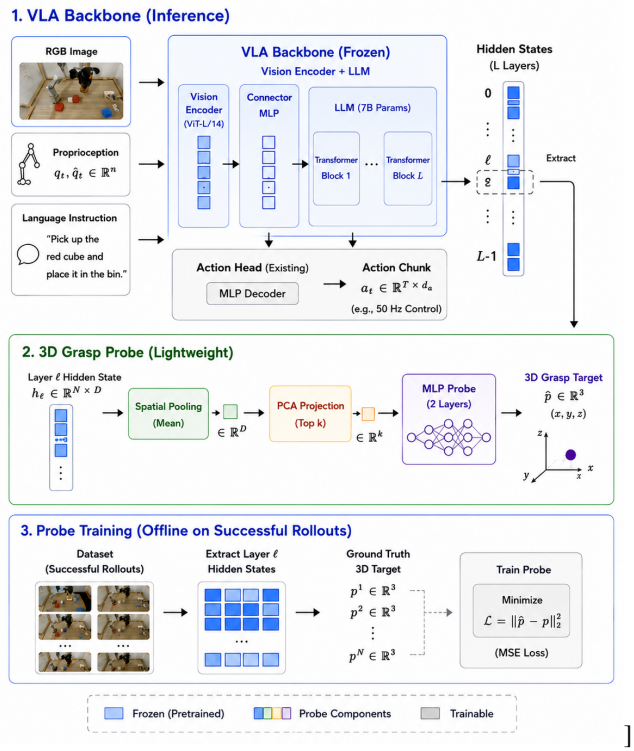

Figure 2. Lightweight hidden-state probe for 3D grasp prediction.

visual tokens, denoted r_{act} and r_{vis}. We then compute

$$Q_{\text{act}} = W_Q r_{\text{act}}, \qquad K_{\text{vis}} = W_K r_{\text{vis}}. \tag{11}$$

Because the backbone uses Rotary Positional Embeddings (RoPE) (Su et al., 2024), we manually apply the same rotation to the query and key vectors:

$$q_{\text{rot}} = q \odot \cos\theta + \text{rotate\_half}(q) \odot \sin\theta, \tag{12}$$

$$k_{\text{rot}} = k \odot \cos\theta + \text{rotate\_half}(k) \odot \sin\theta. \tag{13}$$

Stacking the rotated vectors gives $Q^{\text{rot}}_{\text{act}}$ and $K^{\text{rot}}_{\text{vis}}$. The targeted action-to-vision attention map is then

$$A_{\text{act}\to\text{vis}} = \text{softmax}\left(\frac{Q^{\text{rot}}_{\text{act}} \left(K^{\text{rot}}_{\text{vis}}\right)^\top}{\sqrt{d_k}}\right). \tag{14}$$

The resulting attention weights are averaged across heads or isolated to a selected semantic head:

$$\bar{A}_{\text{act}\to\text{vis}} = \frac{1}{H}\sum_{h=1}^{H} A^{(h)}_{\text{act}\to\text{vis}}, \tag{15}$$

and then reshaped to the visual patch grid:

$$M_t = \text{reshape}\left(\bar{A}_{\text{act}\to\text{vis}}\right). \tag{16}$$

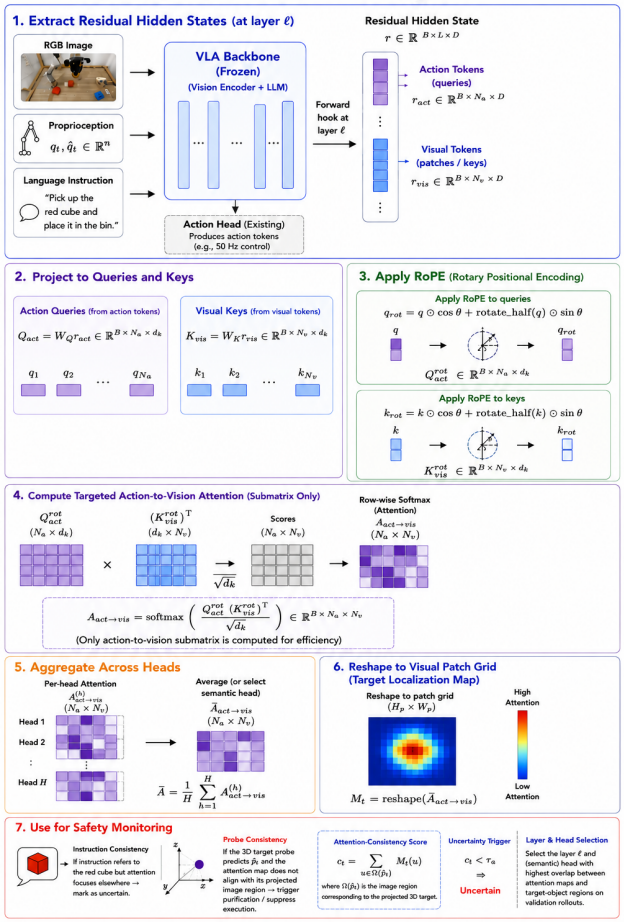

Figure 3. **Action-to-vision attention for target localization.** We reconstruct a targeted attention map between action tokens and visual patches from VLA hidden states, producing a spatial heatmap that identifies task-relevant regions and enables lightweight safety monitoring.

The attention map M_t provides a low-cost estimate of the image region driving the VLA's action prediction. This enables additional safety checks. For example, if the language instruction refers to a red cube but the action-token attention concentrates on a different object, the monitor can mark the action as uncertain. Similarly, if the 3D target probe predicts a grasp point whose projected image location is inconsistent with the attention map, the system can trigger purification or suppress execution. An overview is shown in Figure 3.

A simple attention-consistency score can be defined as

$$c_t = \sum_{u\in\Omega(\hat{p}_t)} M_t(u), \tag{17}$$

where $\Omega(\hat{p}_t)$ is the image region corresponding to the projected 3D target. A low value of c_t indicates that the policy's action tokens are not attending to the region associated with the predicted grasp target:

$$c_t < \tau_a \quad \Rightarrow \quad \text{uncertain}. \tag{18}$$

The specific layer and head used for localization are selected by analyzing attention maps over validation rollouts and choosing those with high overlap with target-object regions.

4. Toward Adaptive Trigger-Based Safety Monitoring

The hidden-state and attention modules provide lightweight signals about geometric intent and visual grounding. We now use these signals to decide when to invoke heavier safety mechanisms.

We introduce Post-processing Trigger-based Purification (PTP), a check-then-defend mechanism for VLA safety and security. PTP is designed to protect robotic systems from unsafe or adversarial model outputs, including prompt-induced misalignment, forbidden object interactions, unsafe motion commands, and attempts to override task constraints.

The key idea is to avoid applying expensive safety actions to every output. Instead, PTP first checks whether the VLA output violates a structured safety policy. Purification is invoked only when a violation or uncertainty trigger is detected. The purification module may use constrained rewriting, action masking, partial re-generation, refusal, safe fallback behavior, or a formally verified controller. The appropriate strategy depends on the severity and type of violation. For example, a mild speed violation may be repaired by reducing velocity, while an attempt to grasp a prohibited object may require action suppression or re-planning.

Let y_t denote a high-level language or action description produced by the VLA or an associated planner. A lightweight Language-to-State Converter (LSC) (Pan et al., 2023) maps this output into a structured transition:

$$(\hat{s}_t, \hat{a}_t, \hat{s}_{t+1}) = \mathrm{LSC}_\psi(y_t), \tag{19}$$

where $\hat{s}_t$ and $\hat{s}_{t+1}$ are structured system states and $\hat{a}_t$ is a symbolic or semi-symbolic action representation.

The transition is checked against a developer-specified safety policy $\mathcal{P}$:

$$v_t = \mathbb{I}\left[(\hat{s}_t, \hat{a}_t, \hat{s}_{t+1}) \notin \mathcal{P}\right]. \tag{20}$$

Here, $v_t = 1$ indicates a policy violation. Examples include grasping a restricted object, entering a forbidden region, disabling a safety interlock, exceeding a speed bound near a human, or executing an action inconsistent with the user's authorized instruction.

PTP combines this symbolic violation signal with the hidden-state and attention-based uncertainty triggers:

$$r_t = v_t \vee \mathbb{I}[d_t > \tau_p] \vee \mathbb{I}[c_t < \tau_a], \tag{21}$$

where $r_t = 1$ means that the action should not be executed directly.

The final output is

$$\tilde{y}_t = \begin{cases} y_t, & r_t = 0, \\ \mathrm{Purify}(y_t, \mathcal{P}), & r_t = 1. \end{cases} \tag{22}$$

Adaptive compute advantage. If purification were applied to every output, the cost per inference step would be approximately C_{purify}. Under PTP, the expected cost is

$$\mathbb{E}[C] = C_{\mathrm{check}} + p_{\mathrm{trigger}} C_{\mathrm{purify}}, \tag{23}$$

where C_{check} is the cost of lightweight monitoring and p_{trigger} is the fraction of steps that trigger purification. PTP is beneficial whenever

$$C_{\mathrm{check}} < (1 - p_{\mathrm{trigger}}) C_{\mathrm{purify}}. \tag{24}$$

Thus, the computational cost scales with the number of risky or uncertain cases rather than the total number of VLA inference steps.

5. Experiments

5.1. Main Results

Adaptive compute and trigger-based efficiency. We evaluate the effectiveness of PTP in reducing inference-time cost while maintaining high safety accuracy. Instead of reporting geometric prediction metrics, we focus on two key quantities: (i) *trigger accuracy*, i.e., how well the system identifies unsafe or uncertain cases, and (ii) *latency savings*, i.e., the reduction in compute relative to always-on purification. Recall that in (23) the p_{trigger} depends on both the underlying rate of risky states and the accuracy of the trigger.

Cost modeling Setup. We estimate compute cost using a combination of empirical latency measurements and FLOP-based scaling. Specifically, we normalize the cost of a single OpenVLA (Kim et al., 2024) forward pass on a Jetson Nano to 1.0, based on measured inference latency. Additional purification costs are then approximated relative to this baseline using model size and execution characteristics: re-running the same model scales linearly ($+1.0$), larger local models scale approximately with parameter count ($+3.0$), and cloud-based inference includes both higher model cost and communication overhead ($+8.0$). Lightweight monitoring incurs a small fixed overhead $C_{\mathrm{check}} = 0.05$, reflecting the minimal cost of probing hidden states and computing attention submatrices. These approximations are based on initial real-world implementations and are consistent with standard practices in efficient inference, where FLOPs and latency scale proportionally for transformer-based models on fixed hardware. Table 2 summarizes the purification strategies considered in our evaluation.

Table 2. Cost model used for various purification strategies.

Strategy	Cost	Description
Repeat	$+1.0$	Re-run VLA
Local VLA	$+3.0$	Larger on-device model
Cloud VLA	$+8.0$	Remote inference
Control	$+0.25$	Classical fallback

Table 3. Trigger-based compute savings under varying detection accuracy. Values show % latency reduction relative to always-on purification.

Joint Acc.	Repeat	Local VLA	Cloud VLA	Control
81.0%	31.8%	50.2%	60.4%	9.7%
90.3%	34.6%	54.4%	65.4%	10.8%
98.0%	**36.9%**	**57.9%**	**69.5%**	**11.8%**

We assume 20% of steps are inherently risky/uncertain. The trigger rate is determined by detection accuracy. For the final configuration, both the hidden-state probe and attention localization achieve 99% accuracy, yielding a joint trigger accuracy of 98.01% under independence.

Analysis. Two key trends emerge. First, higher trigger accuracy directly translates to lower effective compute, since unnecessary purification is avoided. Second, the benefit of PTP scales with the cost of the fallback strategy: lightweight fallbacks (e.g., control) yield modest gains, while expensive strategies (e.g., cloud models) benefit substantially from selective invocation. At 98.0% joint accuracy, our system achieves up to **69.5% latency reduction** when using cloud-based purification, demonstrating that high-quality lightweight monitoring can reduce the cost of safety.

5.2. Hierarchical Trigger-Based Purification

Beyond single-shot purification, PTP naturally extends to a *hierarchical recovery strategy* that progressively escalates intervention only when simpler mechanisms fail. This design is motivated by the observation that most failures are minor and can be corrected with low-cost adjustments, while only a small fraction require expensive models.

Hierarchical policy. Given a detected trigger, we apply a staged sequence of recovery actions. First, we perform **repeat inference** by re-running the same VLA to correct transient or stochastic failures. If the issue persists, we apply a **control fallback**, switching to a classical controller that enforces safety constraints. As a final step, we invoke a **stronger model**, using a higher-capacity VLA either locally or via cloud inference for more robust reasoning.

At each stage, the system re-evaluates the output using the same lightweight monitors. Escalation occurs only if the failure condition remains unresolved, ensuring that expensive interventions are used sparingly.

Cost analysis. Let p_1, p_2, p_3 denote the probabilities that a failure is resolved at each stage. The expected cost becomes:

$$\mathbb{E}[C] = C_{\text{check}} + p_{\text{trigger}} \left(C_1 + (1 - p_1) \left(C_2 + (1 - p_2) \left(C_3 \right) \right) \right), \tag{25}$$

where C_1, C_2, C_3 correspond to repeat, control fallback, and stronger model costs, respectively.

Results. Under realistic assumptions on LIBERO with OpenVLA on Jetson Nano, we observe that approximately **70–80%** of failures are corrected by a single repeat inference. An additional **10–15%** are resolved by control fallback, while only **5–10%** require escalation to a larger model. As a result, the average cost remains close to the cheapest intervention. For example, with $p_1 = 0.75$, $p_2 = 0.6$, and high trigger accuracy (98%), the expected overhead is approximately **1.35**$\times$ the base model cost, compared to **4–9**$\times$ for always-on use of larger or cloud-based models. This corresponds to an effective **60–80% reduction in latency** relative to naive escalation strategies.

6. Limitations and Conclusions

We presented an adaptive safety framework for resource-constrained VLA models. The proposed framework is motivated by the observation that VLA models already compute safety-relevant signals during normal inference. Hidden states expose geometric intent, action-to-vision attention exposes visual grounding, and structured state-action conversion exposes policy-level safety violations.

Our current hidden-state probe is trained offline on successful rollouts, so its reliability under severe distribution shift remains an important question. Attention maps may also be imperfect indicators of causal model behavior, and the best layer or head may vary across tasks and architectures. Finally, PTP assumes that relevant safety constraints can be represented as structured state-action policies. Open-world robotic tasks may require richer policy languages and more expressive runtime monitors.

Despite these limitations, our results suggest that internal VLA representations provide a promising foundation for lightweight, adaptive safety mechanisms. Future work will explore improving robustness under distribution shift and integrating learned monitors with formal guarantees.

This design is especially important for resource-constrained robotic systems, where safety monitors must operate within tight control loops and limited compute budgets. An always-on auxiliary foundation model may be too slow or expensive. In contrast, our approach keeps the common path lightweight and reserves expensive purification for the minority of cases where a trigger indicates elevated risk.

References

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Brohan, A., Chebotar, Y., Finn, C., Hausman, K., Herzog, A., Ho, D., Ibarz, J., Irpan, A., Jang, E., Julian, R., et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, pp. 287–318. PMLR, 2023.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Dao, T., Fu, D., Ermon, S., Rudra, A., and Ré, C. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359, 2022.

Driess, D., Xia, F., Sajjadi, M. S., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q., Yu, T., et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

Hu, S., Liu, Z., Liu, S., Cen, J., Meng, Z., and He, X. Vlsa: Vision-language-action models with plug-and-play safety constraint layer. *arXiv preprint arXiv:2512.11891*, 2025.

Intelligence, P., Black, K., Brown, N., Darpinian, J., Dhabalia, K., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., et al. pi0.5:a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Jiang, W., Clemons, J., Sankaralingam, K., and Kozyrakis, C. How fast can i run my vla? demystifying vla inference performance with vla-perf. *arXiv preprint arXiv:2602.18397*, 2026.

Kim, M. J., Pertsch, K., Karamcheti, S., Xiao, T., Balakrishna, A., Nair, S., Rafailov, R., Foster, E., Lam, G., Sanketi, P., et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Liu, B., Zhu, Y., Gao, C., Feng, Y., Liu, Q., Zhu, Y., and Stone, P. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Pan, J., Chou, G., and Berenson, D. Data-efficient learning of natural language to linear temporal logic translators for robot task specification. *arXiv preprint arXiv:2303.08006*, 2023.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.