

RESCUE: Is This Alarm Worth Acting On?

Counterfactual Evaluation of VLA Runtime Monitors

David Ahmedt-Aristizabal, Abdelwahed Khamis, Mohammad Ali Armin, Lars Petersson
CSIRO, Australia

{David.Ahmedtaristizabal, Lars.Petersson}@csiro.au

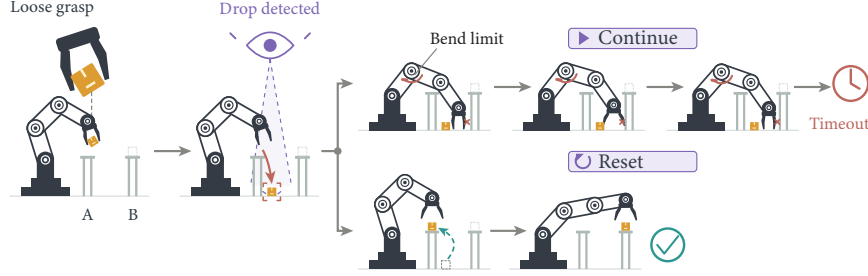


Fig. 1. **Failure prediction does not necessarily imply intervention value.** RESCUE evaluates an existing runtime-monitor alarm by comparing counterfactual interventions from the same robot-policy state. Here, a loose grasp drops the block into a difficult-to-reach location: continuation leads to repeated unsuccessful attempts and timeout, whereas resetting completes the task earlier despite its overhead. Intervention value therefore depends on the available response, alarm timing, false alarms, failure prevalence, and intervention cost.

Abstract—Runtime failure monitors for robot policies are typically evaluated by how accurately they distinguish failed from successful executions. Yet an accurate alarm may arrive too late for recovery, while an unnecessary intervention may disrupt an execution that would otherwise succeed. We introduce RESCUE, a counterfactual framework for evaluating whether a monitor’s alarms enable useful interventions. By restoring simulator and policy states at execution checkpoints, RESCUE compares alternative actions, identifies the latest checkpoint satisfying a specified recovery criterion, and evaluates whether acting on alarms reduces expected task-completion cost. We evaluate RESCUE with OpenVLA-OFT and $\pi_{0.5}$ over LIBERO-Spatial and LIBERO-Plus using 128,340 counterfactual branches. Recoverability depends on the state-intervention pair rather than on robot state alone: at $\tau = 0.5$, 14 of 66 OpenVLA-OFT/LIBERO-Plus failures contain a recoverable checkpoint under continuation, compared with 34 of 66 under continuation or scripted recovery, while recovery reduces recoverability in other conditions. On the primary strong-policy conditions, favourable detection scores do not establish lower intervention cost. Under the evaluated stop-and-reset model, even an idealised detector can afford only a 2.0–3.3% false-alarm rate before unnecessary resets outweigh the savings from stopping failures early. These results show that monitor value depends jointly on the intervention available and the workload in which it is deployed.

I. INTRODUCTION

Vision-language-action (VLA) models are increasingly capable of executing diverse robotic manipulation tasks from visual observations and language instructions [1], [2]. Despite this progress, long-horizon execution remains vulnerable to compounding errors, environmental perturbations, and out-of-distribution conditions. Runtime monitors therefore aim to detect impending or ongoing failures from policy representations, observations, or predictive world models so that execution can be stopped, recovered, replanned, or escalated [3]–[6]. These monitors are commonly selected

using discrimination, calibration, or timing-aware detection metrics.

However, predicting failure and deciding whether to intervene are different problems. A monitor may accurately distinguish failed from successful executions while still providing a poor intervention rule: an alarm may arrive after useful recovery has disappeared, or a false alarm may interrupt an execution that would otherwise succeed. Conversely, an imperfect predictor may still be useful if it alarms while an effective and inexpensive intervention remains available. Related agent studies similarly show that better failure prediction or calibration need not improve outcomes under intervention [7], [8]. In robotics, this distinction is particularly important because interventions alter physical state and may incur reset, delay, or damage costs.

Timing-aware evaluation addresses part of this problem, but elapsed time alone is insufficient. VLA-FAIL [4] incorporates detection time into monitor evaluation, PREFAIL [9] identifies an actionable intervention window for lift-and-place failures, and CheckVLA [5] defines an intervention-relative irreversibility time by testing whether a fixed set of repairs can still restore success. CheckVLA establishes that actionability depends on available repairs, but leaves open how intervention value changes across the trajectory and whether acting remains preferable once false alarms and intervention costs are included [5]. The same robot-policy state can therefore have different value under continuation, recovery, or stop-and-reset.

We introduce **RESCUE** (Recoverability Evaluation of Stopping Criteria Under Execution), a counterfactual framework for evaluating existing runtime monitors as intervention rules. As illustrated in Fig. 1, RESCUE restores simulator and policy states at action-chunk boundaries and evaluates alternative interventions, estimating action-conditioned task

success under a declared intervention repertoire $\mathcal{A}$, execution model ξ , and remaining task budget B . Evaluating these outcomes across the trajectory yields a *recoverability horizon*: the latest checkpoint at which the declared repertoire satisfies a specified recovery criterion. The horizon therefore depends explicitly on the available interventions, execution model, and task budget.

Recoverability establishes what can still succeed, but not whether intervention is worthwhile. Stopping a failed execution can save time, whereas an unnecessary stop incurs a reset and another attempt. RESCUE therefore evaluates interventions in a common cost-to-completion space incorporating failure prevalence, alarm timing, false alarms, and intervention cost. For stop-and-reset, this yields a closed-form break-even false-alarm rate computable from ordinary execution logs, allowing a monitor’s realised operating point to be compared directly with never intervening.

We evaluate RESCUE with OpenVLA-OFT and $\pi_{0.5}$ on LIBERO-Spatial and LIBERO-Plus [10], [11], comprising 1,562 trunk episodes and 128,340 counterfactual branches. At $\tau = 0.5$, only 14 of 66 OpenVLA-OFT/LIBERO-Plus failures contain a recoverable checkpoint under continuation, while 34 of 66 clear under continuation or scripted recovery. Under stop-and-reset, even an idealised detector that identifies every failure immediately can afford only a 2.78–3.32% false-alarm rate under the evaluated cost model. A steps-since-progress monitor tuned to a nominal 5% operating point instead costs +0.0835 reset-equivalents per completed task relative to never intervening. Across matched evaluation sets, conventional detection and timing metrics admit the real monitors in 61 of 75 verdicts, whereas decision value admits none of 15 against never intervening. These results show that failure-prediction quality alone does not establish intervention value. Our contributions are:

- 1) We formulate runtime-monitor evaluation over *state–intervention pairs* and introduce a counterfactual branching protocol that estimates intervention success along robot-policy trajectories, yielding a recoverability horizon conditional on the intervention repertoire, execution model, and remaining task budget.
- 2) We complement recoverability with a decision-value formulation accounting for failure prevalence, alarm timing, false alarms, and intervention cost, and derive a closed-form break-even false-alarm criterion for stop-and-reset from standard execution logs.
- 3) We evaluate the framework at scale across two VLA policies and show that recoverability changes substantially with the available intervention, while monitors favoured by conventional detection and timing metrics can remain decision-negative when evaluated against the intervention they trigger.

II. RELATED WORK

Runtime Failure Monitoring for Robot Policies. Runtime monitors detect execution failures using policy representations, action consistency, and predictive world models. Sentinel [12] pairs temporal action consistency with

VLM-based task-progress reasoning. FIPER [13] combines embedding-space out-of-distribution detection with action-chunk uncertainty, conformally calibrating both scores on successful rollouts. For VLA policies, SAFE [3] predicts failures from policy representations, VLA-FAIL [4] introduces AUCPDT to assess detection accuracy and timing, and Foresight [6] uses action-conditioned world-model latents to detect long-horizon manipulation failures. These works primarily evaluate how reliably and how early failures are identified. CheckVLA [5] couples detection to repair by evaluating suffix rewrites from restored simulator states. It defines an *irreversibility time* beyond which none of its tested repairs restores success, and reports rescue, harm, and task success. RESCUE instead compares alternative interventions throughout execution and evaluates whether acting on alarms lowers expected task-completion cost under the workload’s failure prevalence and intervention costs.

Failure Recovery and Intervention Timing. Recovery methods address how a robot should respond when execution goes wrong. AEGIS [14] selectively escalates to a stronger policy, B2FF [15] selects a pre-imagined familiar future as a recovery goal, and Rewind-IL [16] returns the robot toward a previously verified safe state. PREFAIL [9] identifies the latest actionable intervention time for lift-and-place failures, assessing whether predictions arrive early enough to prevent failure. These works focus on designing recovery behaviour or assessing whether intervention occurs within an actionable window. RESCUE instead evaluates declared branchable interventions from every saved action-chunk boundary, allowing recovery opportunities to be measured throughout execution rather than only at a single trigger point. The recoverability estimate is conditional on the intervention repertoire, execution model, and remaining task budget. RESCUE therefore defines the recoverability horizon as the latest checkpoint satisfying the recovery criterion, without assuming that recoverability decreases monotonically with execution time.

From Prediction Quality to Decision Value. Predictive accuracy does not guarantee beneficial decisions, a distinction reflected in cost-sensitive learning [17] and critiques of aggregate evaluation measures [18]–[20]. ThriftyDagger [21] studies intervention budgets in interactive imitation learning. For LLM agents, Vasudev et al. [7] show that interventions can both rescue failed executions and disrupt successful ones. Zhang et al. [8] measure *intervention advantage* by branching alternative actions from the same execution prefix, showing that better risk calibration need not reduce control regret. RESCUE brings this decision-oriented view to robot manipulation, where interventions alter physical state and resets require another attempt. It evaluates existing monitors by comparing counterfactual interventions from restored robot-policy states and measuring expected cost per completed task under a reset-and-retry model. Rather than computing a model-based reachable set [22], RESCUE estimates action-conditioned recoverability empirically from restored execution states. For stop-and-reset, this yields a break-even false-alarm rate computable from ordinary execution logs.

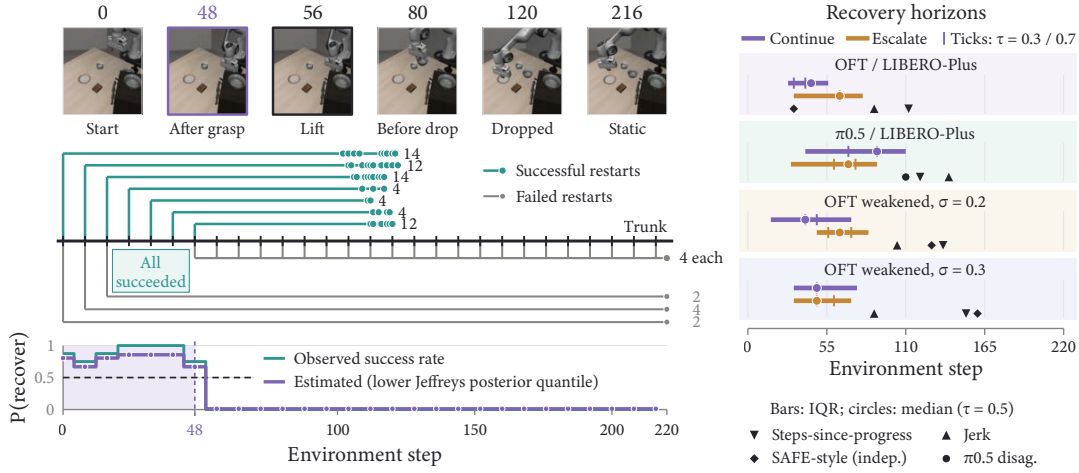


Fig. 2. **Measuring recoverability and its dependence on intervention.** **Left:** Counterfactual restarts from saved checkpoints of a failed OpenVLA-OFT/LIBERO-Plus execution yield observed success rates (teal) and lower-quantile recoverability estimates (purple). At $\tau = 0.5$, step 48 is the last recoverable checkpoint, while step 56 no longer satisfies the recovery criterion. **Right:** Median recoverability horizons (circles), interquartile ranges (bars), and median runtime-monitor alarm times (black symbols) across policy settings. Continue and escalate denote CONTINUE and scripted RECOVER; ticks show median horizons at $\tau = 0.3$ and $\tau = 0.7$. On OpenVLA-OFT/LIBERO-Plus, 14/66 failures contain a qualifying checkpoint under CONTINUE, 33/66 under RECOVER, and 34/66 under their union.

III. METHOD

RESCUE evaluates an existing runtime monitor by asking what each available intervention could have achieved from the robot-policy state at which the monitor alarms. An ordinary policy execution forms a *trunk* trajectory, from which complete robot-policy states are saved at discrete checkpoints. For a checkpoint s_t , RESCUE restores the same state and evaluates alternative interventions through counterfactual continuations. Each state-intervention pair is characterised by two complementary quantities: the probability of completing the current task within the remaining budget, and the expected cost to eventual task completion. The former determines whether a useful intervention remains available, whereas the latter determines whether acting is preferable once reset, delay, and retry costs are considered.

These quantities are conditional on a declared intervention repertoire $\mathcal{A}$, execution model ξ , and remaining task budget B . Evaluating success probability across checkpoints yields an action-conditioned *recoverability horizon*, while evaluating interventions in a common cost space allows monitor alarms to be assessed as decision rules rather than solely as failure predictions. Fig. 2 illustrates how the recoverability horizon is measured on one failed execution by repeatedly restoring checkpoints and observing their counterfactual outcomes.

A. State-Intervention Value and Recoverability

Let π denote the nominal robot policy and let s_t be the complete restorable robot-policy state at checkpoint $t \in \mathcal{T}$. For a branchable intervention a applied at s_t , let ξ denote the execution model governing subsequent stochasticity and let $B(t)$ denote the remaining budget of the current attempt.

We characterise each state-intervention pair through two complementary quantities. First, its within-attempt success probability is

$$Q_{\xi,B}^{\text{succ}}(s_t, a) = \Pr[\text{success within } B(t) \mid s_t, a, \pi]. \quad (1)$$

Second, its expected cost to eventual task completion is

$$Q_{\xi,B}^{\text{cost}}(s_t, a) = \mathbb{E}_{\xi}[C \mid s_t, a, \pi], \quad (2)$$

where C includes the cost incurred by the current attempt and, if needed, subsequent reset-and-retry attempts.

Let $\mathcal{A}$ denote the repertoire of branchable interventions available at the checkpoint. The best success probability and minimum expected cost available from s_t are

$$V_{\mathcal{A},\xi,B}(s_t) = \max_{a \in \mathcal{A}} Q_{\xi,B}^{\text{succ}}(s_t, a), \quad U_{\mathcal{A},\xi,B}(s_t) = \min_{a \in \mathcal{A}} Q_{\xi,B}^{\text{cost}}(s_t, a). \quad (3)$$

For a recovery criterion $\tau \in [0, 1]$, we call s_t *recoverable* when $V_{\mathcal{A},\xi,B}(s_t) \geq \tau$. The corresponding *recoverability horizon* is the latest checkpoint satisfying this criterion,

$$H_{\mathcal{A},\xi,B,\tau} = \max \{t \in \mathcal{T} : V_{\mathcal{A},\xi,B}(s_t) \geq \tau\}, \quad (4)$$

and is undefined when no checkpoint satisfies the criterion.

This formulation makes recoverability a property of the state-intervention pair rather than of the robot state alone. Changing the intervention repertoire, execution model, or remaining budget can therefore change both whether a checkpoint is recoverable and where the horizon lies. For a monitor alarm at checkpoint $\hat{t}$, $V_{\mathcal{A},\xi,B}(s_{\hat{t}})$ describes what remains achievable within the current attempt, while $U_{\mathcal{A},\xi,B}(s_{\hat{t}})$ provides the corresponding decision-oriented view in cost units. The horizon does not assume that recoverability varies monotonically with time.

B. Counterfactual Branching and Estimation

RESCUE estimates the state-intervention quantities in Sec. III-A by repeatedly restoring the same checkpoint and rolling execution forward under a specified intervention. A trunk episode is executed normally while complete robot-policy states are saved at action-chunk boundaries. For checkpoint s_t and branchable intervention $a \in \mathcal{A}$, the state is restored and $K_{t,a}$ seeded counterfactual branches are generated under execution model ξ , each terminating at task

success or when the remaining budget $B(t)$ is exhausted. The same branches provide both success outcomes and realised execution lengths for estimating $Q_{\xi,B}^{\text{succ}}(s_t, a)$ and $Q_{\xi,B}^{\text{cost}}(s_t, a)$.

Let $k_{t,a}$ be the number of successful branches. Because the initial branch count is small, we estimate success probability using a lower Jeffreys posterior quantile rather than the raw fraction,

$$\hat{Q}_{\xi,B}^{\text{succ}}(s_t, a) = Q_q \left[\text{Beta} \left(k_{t,a} + \frac{1}{2}, K_{t,a} - k_{t,a} + \frac{1}{2} \right) \right], \quad (5)$$

where $Q_q[\cdot]$ denotes the q -quantile and $q = 0.25$. Each checkpoint is initially evaluated with $K = 4$ branches; checkpoints in the transition region around the recovery criterion are then expanded to $K = 16$. The estimated repertoire-level recoverability is

$$\hat{V}_{\mathcal{A},\xi,B}(s_t) = \max_{a \in \mathcal{A}} \hat{Q}_{\xi,B}^{\text{succ}}(s_t, a), \quad (6)$$

and the empirical horizon is obtained by substituting $\hat{V}_{\mathcal{A},\xi,B}$ for $V_{\mathcal{A},\xi,B}$ in Eq. (4). The lower posterior quantile biases small- K checkpoint decisions toward dismissal rather than declaring recoverability from a few favourable rollouts.

Counterfactual validity additionally requires a restored checkpoint to reproduce the same experiment as its trunk. RESCUE therefore applies a replay-fidelity check before interpreting branch outcomes: with stochastic perturbations disabled, restoring a deterministic execution must reproduce the corresponding trunk continuation. Fig. 2 illustrates the complete measurement on one failed trajectory, including the seeded branches, raw success fractions, posterior estimate, and resulting horizon.

C. Stop-and-Reset Decision Value

Recoverability determines whether the current attempt can still succeed, but it does not determine whether intervening is preferable to continuing. We therefore evaluate interventions in a common cost-to-completion space. A reset terminates the current attempt, incurs unit cost, and starts a fresh attempt drawn from the same workload. Let c_t denote the execution cost per environment step and let A_π denote the expected cost to eventual completion from such a fresh attempt.

For a branchable intervention $a \in \mathcal{A}$, let L_a be the number of additional steps executed after checkpoint s_t . Its expected cost is

$$Q_{\xi,B}^{\text{cost}}(s_t, a) = c_t(t + \mathbb{E}_\xi[L_a | s_t, a]) + (1 - Q_{\xi,B}^{\text{succ}}(s_t, a))(1 + A_\pi). \quad (7)$$

For immediate stop-and-reset,

$$Q^{\text{cost}}(s_t, \text{stop}) = c_t t + 1 + A_\pi. \quad (8)$$

The already-incurred term $c_t t$ is common to all actions at the same checkpoint and therefore cancels when comparing them. Thus, while Q^{succ} ranks branchable actions by their prospects within the current attempt, Q^{cost} also allows stop-and-reset to be compared on the same scale. All reported decision costs are interpreted as expected cost per completed task under this renewal model.

For stop-and-reset, the comparison with never intervening can be written directly from ordinary execution logs. Let p be the failure prevalence, d the fraction of failures on

which the stopping rule alarms, T_f the mean terminal length of those failures, and t_d their mean alarm time. Let f be the false-alarm rate on otherwise successful episodes, t_{fa} the mean false-alarm time, and $\bar{T}_{s,\text{FA}}$ the mean natural completion length of the successful episodes that are false-alarmed. If C_{never} is the expected cost to completion when never intervening, equating the stopping rule with this baseline gives the break-even false-alarm rate

$$f^* = \frac{p}{1-p} \frac{d c_t (T_f - t_d)}{\Delta + C_{\text{never}}}, \quad \Delta = 1 + c_t (t_{\text{fa}} - \bar{T}_{s,\text{FA}}). \quad (9)$$

The numerator is the expected execution cost saved by stopping detected failures early; the denominator is the excess cost of a false alarm, including the reset and replacement attempt. A realised false-alarm rate below f^* makes the stopping rule preferable to never intervening under the declared cost model, whereas a larger rate makes it decision-negative. Setting $d = 1$ and $t_d = 0$ yields the maximum false-alarm rate affordable by an idealised detector that identifies every failure immediately.

IV. EXPERIMENTS

A. Experimental Setup

Policies, tasks, and corpus. We evaluate RESCUE with OpenVLA-OFT [1] and $\pi_{0.5}$ [2] on LIBERO-Spatial [10] and its LIBERO-Plus perturbations [11]. The complete corpus contains 1,562 trunk episodes, including 262 failures, with 23,430 saved snapshots, 10,083 branched checkpoints, and 128,340 counterfactual branches. Our primary analysis uses the two strong-policy LIBERO-Plus cells, each containing 481 task variants. Two weakened OpenVLA-OFT cells on LIBERO-Spatial (150 episodes each) and two vanilla control cells (200 OpenVLA-OFT and 100 $\pi_{0.5}$ episodes) provide controlled contrasts rather than cross-policy comparisons. All failures in the corpus terminate at the 220-step episode timeout.

Counterfactual interventions. Snapshots are recorded at action-chunk boundaries—every 8 environment steps for OpenVLA-OFT and every 10 for $\pi_{0.5}$ —and restored in a fresh process before branching. We instantiate the branchable repertoire as $\mathcal{A} = \{\text{CONTINUE}, \text{RECOVER}\}$. CONTINUE resumes the nominal policy directly. RECOVER executes a fixed open-loop manoeuvre before resuming the policy: freeze for 8 steps; if the trunk’s recorded gripper command at $t - 1$ is positive, lower for 6 steps and release for 4; then retract for 8 steps and settle for 4. The script therefore lasts 30 steps when this condition holds and 20 otherwise, for both policies. The policy server is idle during the script, and success is credited only after the scripted manoeuvre has finished and nominal policy execution has resumed. Stop-and-reset is not branched and is evaluated analytically using Sec. III-C.

Execution model and recoverability criterion. Unless otherwise stated, counterfactual branches use an action perturbation with $\sigma = 0.05$ on the six continuous arm dimensions; the gripper command and scripted recovery are not perturbed. $\pi_{0.5}$ additionally retains its native policy sampling.

For LIBERO-Plus sensor-noise variants, branches remain conditioned on the corruption realisation observed by the corresponding trunk. The remaining budget is inherited from the original episode, $B(t) = 220 - t$, rather than restarted after intervention. We use $\tau = 0.5$ as the primary recoverability criterion and report sensitivity to $\tau \in \{0.3, 0.7\}$.

Runtime monitors and decision costs. We evaluate four monitor families where applicable: steps-since-progress, action-jerk, sampling disagreement for $\pi_{0.5}$, and a SAFE-style implementation [3] operating on OpenVLA-OFT hidden representations with MLP and LSTM heads. Steps-since-progress counts action-chunk boundaries since the end-effector position last moved by more than 0.01 m between successive boundaries. Action-jerk, a heuristic rather than physical jerk, is the mean absolute first difference between consecutive commanded actions over the trailing 16 steps, averaged over the six continuous arm dimensions. For $\pi_{0.5}$, sampling disagreement is the mean standard deviation of the six arm dimensions over eight independently sampled actions at each boundary. The SAFE-style MLP and LSTM heads use OpenVLA-OFT hidden representations as their input features. For the primary operating point, monitor thresholds are selected on validation data to target a 5% false-alarm rate on successful episodes and are then evaluated on held-out data using their realised test false-alarm rate. For decision-value evaluation, one reset has unit cost and the per-step cost c_t is chosen so that the median successful episode costs 0.1 resets. Confidence intervals for the reported decision costs use 1,000 stratified episode-level bootstrap resamples, with threshold selection repeated within each resample.

B. Recoverability Is Intervention-Relative

We first evaluate whether recoverability is preserved under nominal continuation. As summarised in Fig. 3, where the plotted last recoverable state (LRS) is the empirical recoverability horizon of Eq. (4), at $\tau = 0.5$ only 14 of 66 OpenVLA-OFT/LIBERO-Plus failures contain any checkpoint satisfying the recovery criterion under $\mathcal{A} = \{\text{CONTINUE}\}$; the remaining 52 contain no qualifying checkpoint despite evaluating every saved action-chunk boundary. Among the 14 failures with a defined horizon, the median recoverability horizon is step 44 (IQR 28–56) of the 220-step episode budget. Relative to task-level events, this horizon occurs a median of 7 steps after the first grasp attempt and 34 steps before a subsequent object drop where those events are defined, illustrating that the horizon need not coincide with an obvious visual failure event.

Expanding the available intervention changes the result substantially. Under scripted RECOVER, 33 of the same 66 failures contain a qualifying checkpoint, with a median horizon of step 64 (IQR 32–80). Among the 13 failures with a defined horizon under both interventions, RECOVER shifts the horizon by a median of +40 steps; 11 shifts are positive and two negative (two-sided sign test, $p = 0.022$). With the combined repertoire $\mathcal{A} = \{\text{CONTINUE}, \text{RECOVER}\}$, 34 of 66 failures contain at least one qualifying intervention, with a median horizon of step 64 (IQR 36–80), while

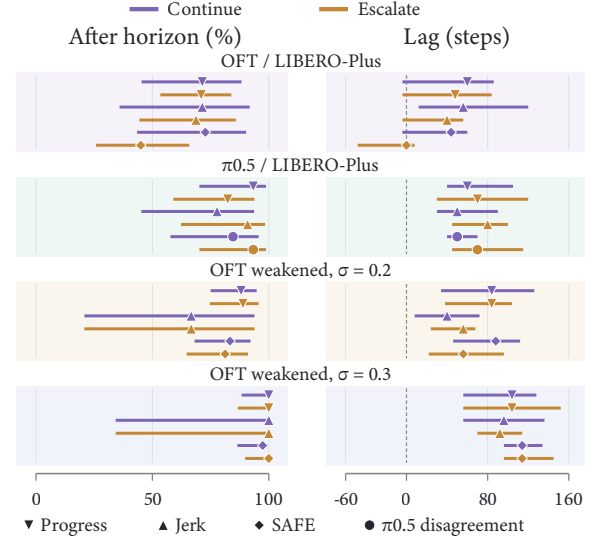


Fig. 3. **Monitor alarms often arrive after the recoverability horizon.** Across four policy settings, **left:** fraction of alarmed failures whose alarm occurs after the horizon, with 95% Wilson confidence intervals; **right:** alarm-minus-horizon lag, with medians and interquartile ranges. Positive lag indicates late detection. Results include failures for which both an alarm and a recoverability horizon are defined under the corresponding intervention. Purple denotes CONTINUE; amber denotes scripted RECOVER, labelled “escalate” in the figure.

32 remain unrecoverable under either intervention. These results directly demonstrate that recoverability depends on the intervention repertoire rather than on the robot-policy state alone.

The effect is not uniformly beneficial across conditions. On $\pi_{0.5}$ /LIBERO-Plus, 25 of 55 failures clear the criterion under each intervention individually, with median horizons of 90 steps for CONTINUE and 70 for RECOVER. On the weakened OpenVLA-OFT cells, scripted RECOVER instead reduces the clearing set from 50/51 to 43/51 at execution noise $\sigma_{\text{exec}} = 0.2$, and from 51/87 to 41/87 at $\sigma_{\text{exec}} = 0.3$. These are within-cell intervention contrasts; we do not interpret the raw differences between OpenVLA-OFT and $\pi_{0.5}$ as a policy comparison.

Finally, the gain from scripted recovery on the primary OpenVLA-OFT cell is not solely an artefact of the injected branch perturbation. For the 61 failures with valid deterministic zero-noise replay, CONTINUE at $\sigma = 0$ produces no successful continuation, whereas scripted RECOVER retains qualifying checkpoints on 31 of those failures. Thus, changing the intervention alters the counterfactual outcome even when the injected branch perturbation is removed.

C. Decision Value of Stop-and-Reset

Recoverability alone does not establish that an alarm should trigger an intervention. We next instantiate the stop-and-reset model of Sec. III-C using the observed failure prevalence, episode durations, alarm times, and realised false-alarm rates. Table I reports the complete break-even worksheet for the steps-since-progress monitor at its validation-selected nominal 5% operating point on the two primary LIBERO-Plus cells.

TABLE I: Break-even stop-and-reset worksheet. Inputs to Eq. (9) for steps-since-progress at its validation-selected nominal 5% false-alarm operating point. $\bar{T}_{s,FA}$ is the mean natural duration of the successful episodes that are actually false-alarmed. The realised false-alarm rate f substantially exceeds the break-even rate f^* on both primary cells.

		OFT-LP	$\pi_{0.5}$ -LP
p	failure prevalence	0.1372	0.1143
T_f	failure length, steps	220	220
$\bar{T}_{s,FA}$	success length, false-alarmed only	116.00	128.87
T_s	median success length, steps	97	112
κ	episode time / reset	0.1	0.1
c_{never}	cost to completion, never intervene	0.2972	0.2551
d	alarm rate on failures	0.879	0.519
t_d	alarm step, failures	133.0	129.3
t_{fa}	alarm step, false alarms	61.9	62.7
f^*	break-even FA rate	1.01%	0.45%
f	realised FA rate	7.25%	7.04%

$\kappa = c_f T_s$ with T_s the median success length. $\bar{T}_{s,FA}$ is the mean length of the successes this rule alarms on, not of all successes; with it the closed form reproduces direct episode accounting exactly.

The available margin for false alarms is small even before accounting for detector imperfections. For an idealised detector with $d = 1$ and $t_d = 0$, Eq. (9) yields an affordable false-alarm rate of only 2.78–3.32% on OpenVLA-OFT/LIBERO-Plus and 2.02–2.39% on $\pi_{0.5}$ /LIBERO-Plus. The range reflects the timing of false alarms and the natural duration of the successful episodes on which they occur. Thus, even a detector that identifies every failure immediately can become decision-negative if it unnecessarily resets only a small fraction of otherwise successful executions.

The idealised ceiling is therefore necessary rather than sufficient. On the weakened $\sigma_{\text{exec}} = 0.3$ cell, for example, steps-since-progress realises a 9.68% false-alarm rate within the 9.59–10.30% perfect-detector ceiling, yet still incurs +0.2613 excess cost per completed task because it neither detects every failure nor alarms at step zero.

For the real steps-since-progress monitor, Table I shows that the available margin is substantially smaller. On OpenVLA-OFT/LIBERO-Plus, the monitor detects a fraction $d = 0.879$ of failures at mean alarm step $t_d = 133.0$, yielding a break-even false-alarm rate of only $f^* = 1.01\%$. Its held-out realised false-alarm rate is instead 7.25%. Direct episode accounting therefore gives an excess cost of +0.0835 reset-equivalents per completed task relative to never intervening (95% CI [0.0294, 0.1392]).

The same pattern holds for $\pi_{0.5}$ /LIBERO-Plus: Table I gives $f^* = 0.45\%$ against a realised false-alarm rate of 7.04%. These results show that a nominal false-alarm target used to select a failure detector need not be economically compatible with the intervention triggered by its alarms.

D. Runtime Monitors as Intervention Rules

We next evaluate the runtime monitors as intervention rules rather than only as failure predictors. The result in Sec. IV-C is not specific to steps-since-progress. At the validation-selected nominal 5% operating point, action-jerk has stop-and-reset excess costs of +0.0563 and +0.0698 reset-equivalents per completed task on OpenVLA-OFT/LIBERO-Plus and $\pi_{0.5}$ /LIBERO-Plus, respectively. On

TABLE II: Detection/timing metrics and intervention value can disagree. Entries report the number of evaluation universes admitting each monitor. Across the real monitors, detection/timing criteria admit 61/75 verdicts, while decision value admits 0/15 against never intervening and 5/15 against rate-matched random alarming. † denotes decoys.

Monitor	u	Detection / timing metrics					Decision value	
		End	Early	< 100	AUCPDT	Timely	vs. Never	vs. Random
Steps-since-progress	4	4/4	1/4	2/4	4/4	2/4	0/4	3/4
Action-jerk	4	4/4	1/4	2/4	4/4	2/4	0/4	1/4
SAFE-style MLP	3	3/3	3/3	3/3	3/3	3/3	0/3	0/3
SAFE-style LSTM	3	3/3	3/3	3/3	3/3	3/3	0/3	0/3
Sampling disagreement	1	1/1	1/1	1/1	1/1	1/1	0/1	1/1
Fixed-step clock†	4	4/4	1/4	3/4	3/4	3/4	1/4	3/4
Label leak†	4	4/4	4/4	4/4	4/4	4/4	4/4	4/4
–Steps-since-progress†	4	0/4	0/4	0/4	0/4	0/4	0/4	4/4
Random alarm†	4	–	–	–	–	0/4	0/4	0/4

OpenVLA-OFT/LIBERO-Plus, the SAFE-style independent head has a three-seed mean excess cost of +0.4988, while the $\pi_{0.5}$ sampling-disagreement monitor is closest to the baseline at +0.0235. None of the evaluated monitor families therefore provides evidence of lower cost than never intervening on the primary strong-policy conditions. This is not a universal no-intervention result. On the higher-prevalence weakened conditions, the SAFE-style monitor has lower cost than never intervening on three of six held-out evaluations; the largest gain is $\Delta C = -0.0431$ (95% CI [−0.0432, −0.0426]) on a cell with 34% failure prevalence.

Changing the intervention triggered by the same alarm can improve the outcome without making the monitor beneficial overall. Routing alarms to scripted RECOVER instead of immediate stop-and-reset reduces excess cost by factors of 3.1–13.4 across the evaluated monitor configurations. Nevertheless, on the two primary strong-policy conditions, no real monitor has a confidence interval establishing lower cost than never intervening under either intervention rule. Thus, the value of an alarm depends on the intervention it triggers, even when the monitor itself is unchanged.

To test whether this result is caused by the nominal 5% operating point, we also select each threshold on validation data by minimising intervention cost. Of the eight monitor-intervention configurations, five select thresholds that produce no alarms on the held-out set. The only negative point estimate, steps-since-progress routed to RECOVER, gives $\Delta C = -0.0058$, but its 95% interval [−0.0204, 0.0143] spans zero. Selecting the operating point directly for intervention cost therefore does not establish an improvement over never intervening.

We next test whether conventional monitor-selection metrics lead to the same verdict. Table II compares the real monitors on matched evaluation universes using three AUROC variants, VLA-FAIL’s AUCPDT [4], a timing criterion, and decision value. *End* is episode-level AUROC using the maximum monitor score over the full trajectory; *Early* restricts that maximum to the task-specific early-exposure window used by SAFE; and *< 100* restricts it to checkpoints before environment step 100. *Timely* is the horizon-relative lead rate at the 5% operating point: the fraction of alarmed failures with a defined recoverability horizon whose alarm occurs at or before that horizon. For this comparison, confidence intervals are computed by resampling the underlying base tasks to account for dependence among LIBERO-Plus

variants. A metric “admits” a monitor when its 95% base-task interval excludes the corresponding reference in the favourable direction; for decision value, this requires lower cost than the reference. Across the 15 real-monitor-universe combinations, the five detection and timing criteria admit the monitors in 61 of 75 possible verdicts. Decision value, in contrast, admits them in 0 of 15 verdicts against never intervening.

The reference used for decision value also affects the verdict. Against a rate-matched random alarm, decision value admits 5 of 15 real-monitor verdicts; however, the deliberately inverted steps-since-progress decoy also clears this reference on all four universes. Outperforming a random alarm is therefore not sufficient to establish useful intervention value. The 61/75 and 0/15 counts are descriptive tallies over related metrics and overlapping evaluation sets, not independent statistical tests.

A sensor-free fixed-step baseline provides another reference. On OpenVLA-OFT/LIBERO-Plus, a validation-selected fixed-step rule stops at step 168 and costs +0.0038 reset-equivalents per completed task (95% CI $[-0.0085, 0.0348]$), compared with +0.0835 for steps-since-progress. Failure discrimination alone therefore does not determine intervention value.

The monitor families fail for different reasons. Steps-since-progress alarms late relative to the recoverability horizon, limiting the execution time that stopping can save. The SAFE-style monitor ranks failures effectively but transfers its nominal operating point poorly: the independent head reaches a three-seed mean realised false-alarm rate of 26.5% at a 5% validation target. Conversely, $\pi_{0.5}$ sampling disagreement realises a 2.3% false-alarm rate—within the idealised ceiling of Sec. IV-C—yet still has positive estimated intervention cost. Discrimination, false-alarm rate, and intervention timing therefore capture different aspects of monitor usefulness.

E. Robustness and Evaluation Controls

Counterfactual fidelity and estimator stability. The counterfactual analysis is meaningful only if restoring a checkpoint reproduces the experiment from which it was saved. Across all six cells, 3,000/3,000 seeded checkpoint replays reproduced the recorded joint trajectory with maximum joint-position deviation 0.0. A stronger zero-noise known-answer test also exposed an observation-channel mismatch on the LIBERO-Plus sensor-noise track; after correcting the branch runner, the affected episodes were re-branched and all results reported here use the corrected labels. We separately re-evaluated 20 OpenVLA-OFT/LIBERO-Plus failures with a fixed $K = 32$ at every checkpoint of both interventions (35,840 branches). At $\tau = 0.5$, the published estimator disagrees with this higher-sampling reference on only 9/560 CONTINUE checkpoints (1.61%) and 6/560 RECOVER checkpoints (1.07%). Whether an episode contains any qualifying checkpoint is unchanged for all 20 failures under both interventions; the estimated horizon moves on 3/20 episodes for CONTINUE and 1/20 for RECOVER. We

therefore interpret the fixed- K experiment as agreement under increased sampling rather than as ground truth. The same fixed- K subset also shows that recoverability need not evolve monotonically with execution time. Against a monotone null, RECOVER is significantly non-monotone on 8/20 episodes, whereas CONTINUE is non-monotone on 0/20 at this sampling level. This supports defining the horizon as the latest qualifying checkpoint without assuming a monotone recoverability boundary.

Decision-cost robustness. Because LIBERO-Plus variants are grouped around ten underlying base tasks, episode-level re-sampling may understate uncertainty when variants from the same task are correlated. We therefore repeated the decision-cost analysis using base-task rather than episode resampling. For steps-since-progress on OpenVLA-OFT/LIBERO-Plus, the 95% interval widens from $[0.0294, 0.1392]$ to $[0.0234, 0.2793]$ but remains above zero. The zero-exclusion verdict is unchanged across the scored monitor configurations. The stop-and-reset conclusion is also not specific to the chosen time price. When a median successful episode is valued at 0.05, 0.1, 0.2, or 0.5 resets, steps-since-progress on OpenVLA-OFT/LIBERO-Plus remains decision-negative, with excess costs of +0.0870, +0.0835, +0.0764, and +0.0551, respectively.

Sensitivity to the recoverability definition. The main conclusions are not tied to a single recovery threshold. For OpenVLA-OFT/LIBERO-Plus under CONTINUE, 18, 14, and 13 of 66 failures contain a qualifying checkpoint at $\tau = 0.3, 0.5$, and 0.7 , respectively, with median horizons at steps 40, 44, and 32. The clock convention can also change the answer: replacing the inherited deadline with a fresh budget equal to the median successful-episode length shifts the paired $\pi_{0.5}$ /LIBERO-Plus horizon by +30 steps (IQR 0–55) on the 27 failures labelled under both conventions and changes 29/55 episode labels. Because this auxiliary clock ablation was run before the LIBERO-Plus sensor-noise branch correction, we use it only to assess sensitivity to B ; all primary horizon results use the corrected labels.

Timing requires a matched null. A monitor alarming after the recoverability horizon does not by itself establish poor timing because the horizon often closes early in these trajectories. A uniformly random alarm also occurs after the horizon on approximately 73% of recoverable failures in each OpenVLA-OFT cell. Indeed, the OpenVLA-OFT/LIBERO-Plus SAFE-style timing rates fall within the corresponding random-alarm interval. We therefore compare timing metrics with alarms generated on the same episode population rather than treating zero lag as an implicit null. Nevertheless, under CONTINUE, every real monitor alarms after the horizon in more than half of the episodes for which both quantities are defined.

Timeout-length confounding. All 262 failures in the corpus are 220-step timeouts, creating a strong length cue for episode-level detection metrics. A score using only the number of observed action-chunk boundaries attains AUROC 1.000 on all 18 validation splits and again on all three task-disjoint holdouts. Using a per-task early-exposure control re-

duces the steps-since-progress AUROC from 0.979 to 0.637, demonstrating that uncorrected episode-level discrimination can substantially reflect trajectory length rather than failure information. We therefore treat the conventional detection metrics in Table II as monitor-selection diagnostics, not as substitutes for intervention-value evaluation.

V. DISCUSSION AND LIMITATIONS

RESCUE shifts runtime-monitor evaluation from the monitor alone to the monitor–intervention pair. A useful alarm must arrive while an effective intervention remains available and while acting is preferable under the prevalence and costs of the intended workload. For stop-and-reset, Eq. (9) makes this comparison directly from ordinary execution logs, using the monitor’s realised held-out false-alarm rate rather than its nominal target. Recoverability is likewise conditional on the declared repertoire $\mathcal{A}$, execution model ξ , and remaining budget B ; it should therefore not be interpreted as an intrinsic property of a robot state or policy. A different recovery controller, deadline, execution model, or cost structure can change both the recoverable set and the preferred intervention. Here we evaluate only CONTINUE and one fixed scripted RECOVER, and make no claim about richer or learned recovery repertoires.

Several limitations bound the empirical conclusions. All experiments use MuJoCo and LIBERO/LIBERO-Plus, and all 262 observed failures are timeouts; hardware may introduce contact dynamics, damage, latency, and recovery costs absent from this benchmark. The branch perturbation and decision costs are declared rather than calibrated from deployment. We evaluate two VLA policies and four monitor families, without training a new detector. The lightweight-monitor splits are episode-disjoint but not base-task-disjoint, so their false-alarm rates do not establish generalisation to unseen tasks. Base-task uncertainty estimates also use few clusters—ten overall and three held-out base tasks per SAFE-style seed—and may therefore be optimistic. The two LIBERO-Plus policy cells fail on different perturbation tracks, precluding direct policy comparison. Finally, the fixed- K analysis in Sec. IV-E supports episode-level stability but does not make the estimated horizon noise-free.

VI. CONCLUSION

We introduced RESCUE, a counterfactual framework for evaluating runtime monitors together with the interventions their alarms trigger. Rather than treating failure prediction as sufficient, RESCUE measures action-conditioned recoverability from restored robot-policy states and evaluates interventions in a common cost-to-completion space. Across 1,562 VLA episodes and 128,340 counterfactual branches, we find that recoverability changes with the available intervention and that monitors favoured by conventional detection and timing metrics can remain decision-negative once false alarms, timing, prevalence, and intervention cost are included. These results support evaluating runtime monitors together with the interventions they trigger, rather than relying on failure-prediction quality alone.

REFERENCES

- [1] M. J. Kim, C. Finn, and P. Liang, “Fine-tuning vision-language-action models: Optimizing speed and success,” *arXiv preprint arXiv:2502.19645*, 2025.
- [2] K. Black, *et al.*, “ $\pi_{0.5}$: a vision-language-action model with open-world generalization,” in *Proceedings of The 9th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 305, 2025, pp. 17–40.
- [3] Q. Gu, Y. Ju, S. Sun, I. Gilitschenski, H. Nishimura, M. Itkina, and F. Shkurti, “Safe: Multitask failure detection for vision-language-action models,” *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 38, pp. 40 041–40 076, 2025.
- [4] F. Seligmann, E. Gospodinov, E. U. Dincer, and G. Neumann, “Vla-fail: Efficient task failure detection for finetuned vision-language-action models,” *arXiv preprint arXiv:2606.21386*, 2026.
- [5] Y. Liu, *et al.*, “Checkvla: Execution-time verification with action-conditioned world model for long-horizon mobile manipulation,” *arXiv preprint arXiv:2607.26789*, 2026.
- [6] H. Zhang, Y. Lu, B. Wang, X. Kang, Y.-L. Kuo, Z. Cheng, M. Wang, and O. C. Jenkins, “Foresight: Failure detection for long-horizon robotic manipulation with action-conditioned world model latents,” *arXiv preprint arXiv:2606.23085*, 2026.
- [7] R. Vasudev, M. Russak, D. Bikel, and W. Alshikh, “Accurate failure prediction in agents does not imply effective failure prevention,” *arXiv preprint arXiv:2602.03338*, 2026.
- [8] C. Zhang, Z. Wan, X. Yu, J. Wu, Q. Wen, P. Zhou, W. Zhao, and I. Tsang, “Calibration is not control: Why llm-agent oversight needs intervention,” *arXiv preprint arXiv:2606.21399*, 2026.
- [9] Z. Shanguan, R. Chitale, R. Patel, S. K. Gupta, and D. Seita, “Prefail: Identifying precursors to failures in robotic lift-and-place tasks to improve task execution performance,” *arXiv preprint arXiv:2607.16921*, 2026.
- [10] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, “Libero: Benchmarking knowledge transfer for lifelong robot learning,” *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, pp. 44 776–44 791, 2023.
- [11] S. Fei, *et al.*, “Libero-plus: In-depth robustness analysis of vision-language-action models,” *arXiv preprint arXiv:2510.13626*, 2025.
- [12] C. Agia, R. Sinha, J. Yang, Z. Cao, R. Antonova, M. Pavone, and J. Bohg, “Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress,” in *Proceedings of The 8th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 270. PMLR, 06–09 Nov 2025, pp. 689–723.
- [13] R. Römer, A. Kobras, L. Worbis, and A. Schoellig, “Failure prediction at runtime for generative robot policies,” *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 38, pp. 7631–7670, 2025.
- [14] J. Chen, “AEGIS: A backup reflex for physical ai,” *arXiv preprint arXiv:2606.06660*, 2026.
- [15] S. Shin, J. Kim, H. Park, H. Kim, H. Lee, H.-S. Kim, and B.-T. Zhang, “Back to the familiar future: Failure recovery for vla policies via pre-imagined milestone selection,” *arXiv preprint arXiv:2606.09258*, 2026.
- [16] G. Zheng, S. Seenivasan, M. Johnson-Roberson, and W. Zhi, “Rewind-il: Online failure detection and state respawning for imitation learning,” *arXiv preprint arXiv:2604.16683*, 2026.
- [17] C. Elkan, “The foundations of cost-sensitive learning,” in *International joint conference on artificial intelligence*, vol. 17, no. 1. Lawrence Erlbaum Associates Ltd, 2001, pp. 973–978.
- [18] A. J. Vickers and E. B. Elkin, “Decision curve analysis: a novel method for evaluating prediction models,” *Medical Decision Making*, vol. 26, no. 6, pp. 565–574, 2006.
- [19] C. Drummond and R. C. Holte, “Cost curves: An improved method for visualizing classifier performance,” *Machine learning*, vol. 65, no. 1, pp. 95–130, 2006.
- [20] D. J. Hand, “Measuring classifier performance: a coherent alternative to the area under the roc curve,” *Machine learning*, vol. 77, no. 1, pp. 103–123, 2009.
- [21] R. Hoque, A. Balakrishna, E. Novoseller, A. Wilcox, D. S. Brown, and K. Goldberg, “Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning,” in *Proceedings of the 5th Conference on Robot Learning*, vol. 164, 2022.
- [22] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, “Hamilton-jacobi reachability: A brief overview and recent advances,” in *2017 IEEE 56th annual conference on decision and control (CDC)*. IEEE, 2017, pp. 2242–2253.