DART: PROACTIVE RUNTIME FAILURE DETECTION FOR VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advances in Vision-Language-Action (VLA) models have enabled strong generalization across diverse robotic manipulation tasks, yet their real-world deployment remains limited by the lack of reliable runtime safety mechanisms. Existing failure detection methods typically rely on coarse trajectory-level supervision or reactive failure scoring, limiting their ability to identify subtle early-stage deviations and rapidly emerging failures. We propose DART, a proactive runtime failure detection framework for VLA policies that jointly addresses sudden failures and cumulative long-horizon deviations through a dual-horizon architecture: a short-horizon module leverages internal semantic and action representations from the VLA to capture abrupt behavioral inconsistencies, while a long-horizon module utilizes a video generation world model to anticipate future state drift before failures fully manifest. To provide fine-grained supervision, we further introduce an automated VLM-based annotation pipeline that generates timestep-level failure labels from execution trajectories while reducing the need for dense manual annotation. Extensive experiments across simulated and real-world manipulation environments on multiple VLA policies demonstrate that DART achieves a favorable trade-off between detection accuracy and temporal responsiveness, particularly under unseen tasks.

1 INTRODUCTION

Recent advances in Vision-Language-Action (VLA) models have demonstrated remarkable capabilities as generalist policies for robotic manipulation, enabling robots to interpret diverse language instructions and generalize across a wide range of tasks and environments (Kim et al., 2024; Black et al., 2024; Pertsch et al., 2025; Intelligence et al., 2025b; Ghosh et al., 2024; Gu et al., 2025a; Apanasevich et al., 2026). However, unlike conventional rule-based methods that operate within well-defined and predictable boundaries, such end-to-end learned policies remain prone to operational errors and semantic deviations when encountering out-of-distribution (OOD) scenarios, dexterous manipulation demands, and complex dynamic environments—posing substantial safety risks during real-world deployment (Black et al., 2024; Intelligence et al., 2025b). Accurate and fast failure detection during VLA policy execution is therefore important for improving the safety and robustness of robotic systems.

Since robotic datasets typically lack fine-grained failure annotations, several recent works (Agia et al., 2024; Xu et al., 2025; Römer et al., 2025; Zhou et al., 2026; Gu et al., 2025b) resort to unsupervised or semi-supervised paradigms from the broader machine learning community for failure detection, identifying failures by comparing positive and negative trajectory distributions within a learned high-dimensional feature space. While these methods can be effective and achieve real-time failure detection at runtime, they lack explicit multi-modal semantic reasoning and thus struggle to generalize across diverse failure modes and environments. Moreover, their reliance on accumulated failure scores exceeding a predefined threshold can make them insensitive to subtle early-stage deviations, introducing detection latency in scenarios demanding rapid intervention.

To address these limitations, we study **proactive runtime failure detection**, aiming to identify emerging failure risks before execution failures explicitly manifest. We focus on two representative temporal patterns of execution-time failures in VLA policies: *sudden failures*, caused by critical short-term errors (e.g., object slippage during grasping) that trigger abrupt state transitions, and

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.

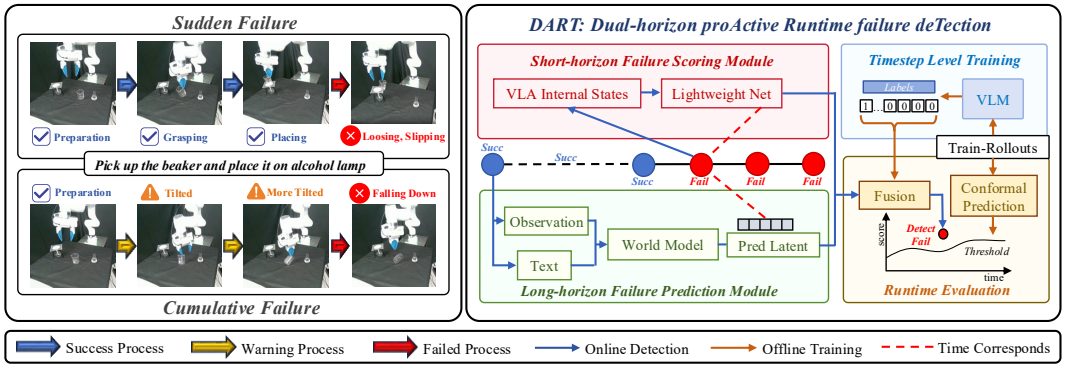


Figure 1: *Left*: Two failure types in VLA manipulation tasks. *Right*: Our dual-horizon runtime failure detection framework to achieve accurate and timely detection for the two failure types, including two detection modules, as well as training and evaluation components.

cumulative failures, arising from compounding multi-step errors that gradually deviate from the intended execution trajectory. These two modes exhibit fundamentally different temporal signatures and thus demand distinct detection strategies: sudden failures require effectively modeling short-horizon robot and environment states to capture abrupt changes, whereas for cumulative failures, per-step deviations can be negligibly small, making it beneficial to anticipate the long-term consequences of the current state. Notably, proactive detection under both modes requires timestep-level failure annotations rather than the coarse trajectory-level labels used in prior work, as the detector must learn fine-grained temporal transitions between normal execution and emerging failure states to enable timely intervention.

To this end, we propose **DART**, a *Dual-horizon framework for proActive Runtime failure deTection*. DART comprises two complementary detection modules designed to capture failure signals at different temporal horizons, shown in Fig. 1. The short-horizon module jointly exploits semantic representations from the VLA's multimodal backbone and control representations from its action expert to capture fine-grained state transitions and rapidly emerging behavioral inconsistencies. Complementarily, the long-horizon module projects the current state into future latent representations to anticipate progressively developing failure risks before they fully manifest. The two modules produce complementary scores that are fused at each timestep, with the decision threshold calibrated via conformal prediction. This design assigns complementary temporal roles to the two detection horizons, which we further analyze through failure-type-specific evaluations on sudden and cumulative failures. To obtain the fine-grained supervision required by our framework, we further develop an automated VLM-based annotation pipeline to generate timestep-level failure labels, enabling temporally aligned supervision for the short and long horizon modules while reducing the need for costly dense manual annotation. We conduct extensive experiments on both simulation benchmarks and real-world manipulation tasks, evaluating state-of-the-art VLA policies including π_0 and $\pi_{0.5}$. Across all settings, DART achieves a favorable balance between detection accuracy and temporal responsiveness. We further analyze detection timing distributions and different failure types, showing the complementary roles of short-horizon failure perception and long-horizon failure anticipation across heterogeneous temporal failure patterns.

The main contributions are as follows: (1) We introduce a dual-horizon runtime failure detection framework that combines VLA internal semantic and control representations for short-horizon failure perception with world-model-predicted latents for long-horizon failure anticipation. (2) We propose an automated VLM-based timestep-level failure label generation pipeline that provides fine-grained temporal supervision for proactive failure detection. (3) We conduct extensive evaluations across two VLA policies in simulation and real-world environments, including analyses of detection accuracy, temporal error distributions, failure types, and supervision horizons, demonstrating the complementary behaviors of the proposed detection horizons.

2 Related Works

Vision-Language-Action Models Vision-Language-Action (VLA) models unify visual perception, language understanding, and robot control within an end-to-end framework. Recent works (Kim

et al., 2024; Black et al., 2024) leverage action discretization or Flow Matching (Liu, 2022; Lipman et al., 2023) to directly map multimodal observations to robot control sequences. In particular, the π series (Intelligence et al., 2025b;a; 2026) has further improved trajectory stability, open-world generalization, and continual learning. However, VLA policies remain prone to failures under out-of-distribution (OOD) scenarios, dexterous manipulation demands, and complex dynamic environments, motivating runtime failure detection before failures fully manifest.

Failure Detection Methods for Robot Policy Existing failure detection methods for robotic policies mainly fall into two categories: distribution-based methods and methods leveraging external prior knowledge.

Distribution-based methods model observations or actions in successful and failed states and detect failures based on distributional deviations (Agia et al., 2024; Xu et al., 2025; Römer et al., 2025; Zhou et al., 2026; Gu et al., 2025b). Many rely on externally extracted visual or action features, which may provide limited task-level semantics for distinguishing failures from benign distribution shifts. Recent work exploits VLA internal diffusion features (Gu et al., 2025b), while complementary semantic information encoded by multimodal VLA representations remains underexplored. Moreover, episode-level supervision provides limited temporal guidance for associating timestep-level features with evolving execution states, potentially reducing sensitivity to early failure deviations and increasing detection latency.

The second category leverages VLM/LLMs or world models. VLM/LLM-based approaches (Yang et al., 2026; Grislain et al., 2025; Ganai et al., 2025; Wu et al., 2025; Duan et al., 2025; Sinha et al., 2024) use prompting or fine-tuning for failure reasoning, but can be sensitive to incidental visual fluctuations, depend on prompt or training-data coverage, and incur substantial runtime inference costs. Existing world-model-based methods (Liu et al., 2024; Grislain et al., 2025) classify failures from predicted latent representations, where prediction errors under complex or unseen dynamics can propagate to failure estimates. Such approaches also often rely on coarse supervision for associating predicted latents with success or failure states.

More importantly, existing methods typically operate at a single temporal scale, either reacting to current deviations or assessing future risk through predictive representations. Such single-horizon treatment overlooks heterogeneous failure dynamics: abrupt failures benefit from immediate perception, while gradually accumulated deviations benefit from anticipatory reasoning. DART instead integrates short-horizon failure perception and long-horizon failure anticipation within a unified runtime detection framework.

Video Generation World Model World models learn compact latent representations of environment dynamics for internal prediction and planning. Early latent dynamics models (Hafner et al., 2019) and the Dreamer series (Hafner et al., 2023) established this paradigm, while recent video generation world models (Wang et al., 2025; Qin et al., 2025; Bruce et al., 2024) demonstrate strong capabilities in modeling temporally extended visual dynamics. Their predicted latent representations capture how the current scene may evolve over time, providing anticipatory information about failure risks before they become directly observable. This motivates our use of predicted video latents for long-horizon failure prediction.

3 METHODS

3.1 PROBLEM FORMULATION

During execution, a VLA at planning step t processes observations o_t and text l_t to output an action chunk $A_t = [a_t, \ldots, a_{t+H-1}]$. After executing the first H' actions, the policy replans at $t + H'$. We propose a dual-horizon framework for failure detection. The short-horizon module leverages VLA internal features e_t to detect abrupt transitions via $s_t^{\text{short}} = f_\phi(e_t)$, while the long-horizon module employs a world model to predict future latent states $\hat{z}_{t+1:t+\Delta}$ for anticipating cumulative deviations via $s_t^{\text{long}} = g_\theta(\hat{z}_{t+1:t+\Delta})$. The fused failure score is $s_t = h_\lambda(s_t^{\text{short}}, s_t^{\text{long}})$. When $s_t > \delta$, which is calibrated via conformal prediction, the system raises an alarm for intervention.

For each failed trajectory, we define the failure manifestation time t_{fail} as the earliest timestep at which the execution failure becomes explicitly and unambiguously observable, as identified by human

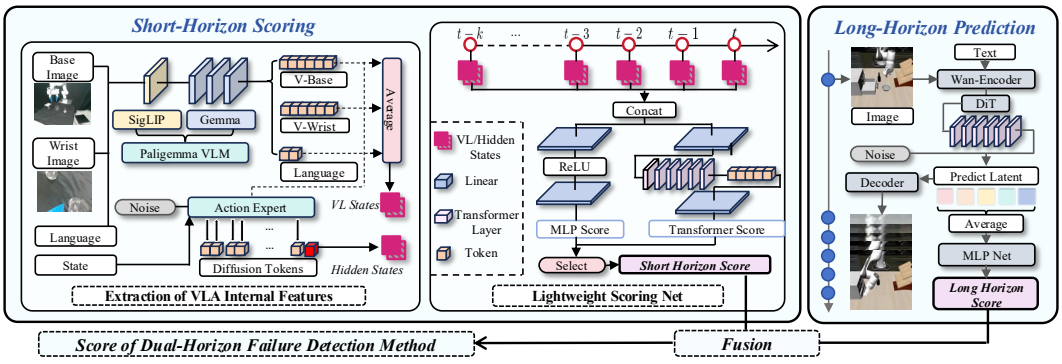


Figure 2: Pipeline of the proposed dual-horizon failure detection framework. *Left*: Our short-horizon scoring module, including feature extraction and scoring net; *Right*: Our long-horizon prediction module; We combine the two to serve as the final score.

annotators. We compare the first alarm time t_{alarm} against t_{fail} to characterize the temporal behavior of the detector: $t_{\text{alarm}} < t_{\text{fail}}$ indicates proactive detection before the failure explicitly manifests, whereas $t_{\text{alarm}} > t_{\text{fail}}$ indicates delayed detection.

3.2 Short-Horizon Failure Scoring Module

As shown in Fig. 2, to capture both semantic state transitions and control-level inconsistencies, at each replanning step t, we extract two complementary internal representations from the VLA policy. First, from the VLM backbone (Beyer et al., 2024), we obtain cross-modally fused Vision-Language (VL) states $\mathbf{z}_t \in \mathbb{R}^{6144}$ by concatenating the base camera, wrist camera, and instruction features $(\mathbf{v}_{base}, \mathbf{v}_{wrist}, \mathbf{l} \in \mathbb{R}^{2048})$, each of which implicitly encodes inter-modal semantic alignment. Second, we extract the 1024-dimensional Hidden Control State $\mathbf{h}_t$ from the final denoising step of the policy's action expert. This feature encodes the model's immediate control intent and action-generation dynamics, facilitating the detection of subtle control-level failures.

To capture dynamic behavioral progressions, we process temporal windows $\mathcal{W}_t = \{(\mathbf{h}_i, \mathbf{z}_i)\}_{i=t-k}^{t}$ using two architectural variants: an MLP-based network with residual connections for point-wise failure mapping, and a Transformer-based architecture that utilizes self-attention to identify subtle shifts through temporal correlations within the local observation window. The VL and hidden-state features are processed by independent detection heads and fused into a short-horizon failure score $s_t^{\text{short}} \in [0, 1]$. The short-horizon module is optimized with $\mathcal{L}_{\text{short}} = \sum_t \mathbb{1}(\text{valid}) \cdot \|s_t^{\text{short}} - Y_t\|^2$, where Y_t denotes the timestep-level failure label at the current timestep t.

3.3 Long-Horizon Failure Prediction Module

As shown in Fig. 2, at each planning step t, the fine-tuned WAN model (Wang et al., 2025) generates a sequence of future latent features $\hat{z}_{t+1:t+\Delta}$ through temporal diffusion inference conditioned on the current visual observation, task instruction, and execution progress. These predicted features capture the anticipated high-level spatiotemporal evolution of the environment, allowing the long-horizon module to identify cumulative deviations that may not be immediately observable from the current policy state. To aggregate the predicted sequence into a stable representation for downstream failure prediction, we apply mean pooling over the temporal dimension, resulting in $\bar{z}_t = \frac{1}{\Delta} \sum_{i=1}^{\Delta} \hat{z}_{t+i}$. The resulting feature $\bar{z}_t$ provides a compact representation of the anticipated future dynamics while mitigating noise from individual predicted frames.

The aggregated latent representation is then fed into an MLP-based network to estimate the long-horizon failure score $s_t^{\text{long}} = g_\theta(\bar{z}_t)$. Unlike the short-horizon module, which focuses on detecting failures from the current policy state, the long-horizon module is designed to anticipate future failure risks. Therefore, during training, the score predicted at timestep t is supervised using the failure label at a future timestep $t + n$, encouraging the model to capture emerging failure trends before they explicitly manifest. The corresponding objective is $\mathcal{L}_{\text{long}} = \sum_t \mathbb{1}(\text{valid}) \cdot \|s_t^{\text{long}} - Y_{t+n}\|^2$, where Y_{t+n} denotes the failure label at the future timestep $t + n$. This horizon-aligned supervision

enables the long-horizon module to complement the short-horizon module, particularly for failures that develop progressively over time.

Finally, the short and long horizon scores are combined through weighted fusion $s_t = \lambda s_t^{\text{short}} + (1 - \lambda) s_t^{\text{long}}$, where the two branches provide complementary estimates of immediate and anticipated failure risks, respectively.

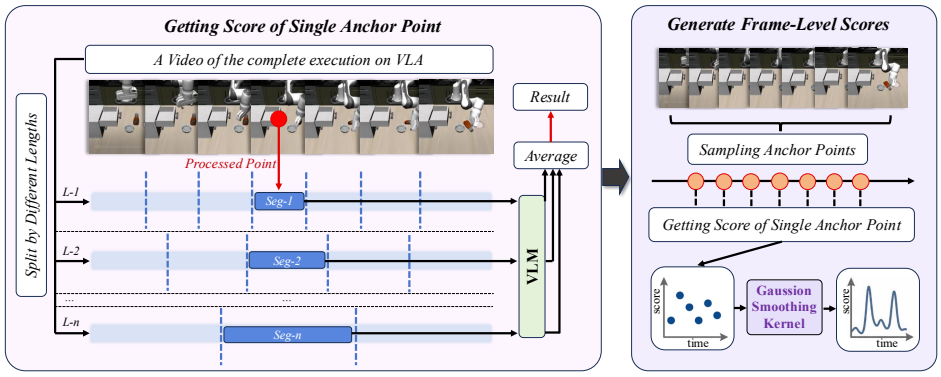


Figure 3: Pipeline of the proposed timestep-level failure label generation process. We first obtain VLM-based scores at sparsely sampled anchor points through multi-scale temporal segmentation, and then apply temporal smoothing to generate dense timestep-level failure scores.

3.4 Timestep-Level Failure Label Generation

Training the dual-horizon detector requires fine-grained temporal supervision, while dense timestep-level annotations are costly to obtain manually. As shown in Fig. 3, we propose an offline VLM-based labeling pipeline (Team, 2025) to automatically generate timestep-level failure labels from execution videos. Given a video $\mathcal{V}$, we partition it into segments $\mathcal{S}_i = \{s_{i,1}, \ldots, s_{i,n_i}\}$ at multiple temporal scales $\mathcal{L} = \{l_0, \ldots, l_k\}$, where each segment $s_{i,j}$ is assigned a failure score $f(s_{i,j}) \in [0, 1]$ via VLM inference. At sparsely sampled anchor points $\mathcal{T} = \{t_0, t_1, \ldots\}$, we aggregate predictions across temporal scales as $\hat{A}_t = \frac{1}{|\mathcal{L}|} \sum_{i=0}^{k} f(s_{i,\text{idx}(t,i)})$, $\text{idx}(t, i) = \lfloor t/l_i \rfloor$. We then apply Gaussian kernel smoothing (Ma et al., 2025) to obtain dense timestep-level scores $A_\tau = \frac{\sum_{t \in \mathcal{T}} \hat{A}_t \exp(-(\tau - t)^2 / 2\sigma^2)}{\sum_{t \in \mathcal{T}} \exp(-(\tau - t)^2 / 2\sigma^2)}$, where σ controls the temporal smoothness. The resulting timestep-level scores are directly used as timestep-level supervision, i.e., $Y_t = A_t$, for training the short and long horizon modules.

4 Experiments

4.1 Environments, Baselines and Evaluation Protocol

We evaluate on 4 LIBERO suites (Liu et al., 2023)—**10**, **90**, **Goal**, and **Spatial**, using trajectories collected with π_0 (Black et al., 2024) and $\pi_{0.5}$ (Intelligence et al., 2025b), with timestep-level labels generated by our VLM-based annotation pipeline. For real-world evaluation, we use a Franka robot equipped with an Agentview (Realsense D455) and a Wristview (Realsense D435), shown in Fig. 8. We collect 200 execution sequences using a fine-tuned $\pi_{0.5}$ policy for training and evaluation.

We compare our framework against several advanced baselines: (i) **STAC (Agia et al., 2024)** detects failures from inconsistencies between overlapping action chunks. (ii) **SAFE (Gu et al., 2025b)** predicts failure probabilities from VLA internal hidden states. (iii) **Fail-Detect (Xu et al., 2025)** models success distributions using flow matching and detects distributional deviations. (iv) **RND (He et al., 2024)** detects failures using prediction errors between a learned predictor and a fixed random target network. For Fail-Detect and RND, we additionally model the failure distribution.

We calibrate the decision threshold using Conformal Prediction (Shafer & Vovk, 2008; Law, 2006) on successful validation trajectories. The normalized temporal error for trajectory i is defined as $e_i = (t_{\text{alarm},i} - t_{\text{fail},i})/T_i$, where $t_{\text{alarm},i}$ is the first alarm time, $t_{\text{fail},i}$ is the human-annotated failure manifestation time defined in Sec. 3.1, and T_i is the rollout length. We evaluate detection

Table 1: Quantitative comparison of failure detection performance in simulation. We report five metrics on seen and unseen tasks for both π_0 and $\pi_{0.5}$ policies.

Policy	Method	Recall ↑	Precision ↑	Bal. Acc. ↑	DT ↓	AT ↓
			π_0 – Seen Tasks			
π_0	Fail-Detect (T)	0.9308	0.8040	0.8360	0.0588	0.1834
	RND (T)	0.9192	0.7071	0.7425	0.1312	0.1925
	SAFE	0.8962	0.8292	0.8428	0.0882	0.2621
	STAC	0.7385	0.8205	0.7771	0.0842	**0.1538**
	DART (Short)	0.9192	**0.9192**	0.9136	0.0843	0.2073
	DART (Long)	0.9462	0.7910	0.8305	**0.0234**	0.2606
	DART (Full)	**0.9769**	0.8912	**0.9205**	0.0413	0.2354
			π_0 – Unseen Tasks			
π_0	Fail-Detect (T)	**0.9320**	0.7328	0.8309	0.0530	0.2251
	RND (T)	0.9029	0.5849	0.6966	0.2030	0.2974
	SAFE	0.6990	0.7579	0.7607	0.1626	0.4403
	STAC	0.6262	0.6649	0.6876	0.0990	**0.1713**
	DART (Short)	0.8835	0.7948	**0.8510**	0.0527	0.2001
	DART (Long)	0.7379	**0.8492**	0.8168	**0.0002**	0.3832
	DART (Full)	0.8495	0.8065	0.8437	0.0125	0.2876
			$\pi_{0.5}$ – Seen Tasks			
$\pi_{0.5}$	Fail-Detect (T)	**0.9919**	0.8079	**0.9036**	0.2501	0.3090
	RND (T)	0.8943	0.5759	0.6892	0.2740	0.3217
	SAFE	0.9593	0.6592	0.7854	**0.0574**	**0.1870**
	STAC	0.6748	**0.8218**	0.7801	0.3281	0.3815
	DART (Short)	0.9756	0.8163	0.9018	0.1151	0.2390
	DART (Long)	0.8862	0.7415	0.8221	0.1757	0.2733
	DART (Full)	0.9837	0.7806	0.8836	0.1099	0.2233
			$\pi_{0.5}$ – Unseen Tasks			
$\pi_{0.5}$	Fail-Detect (T)	0.9697	0.7665	0.8688	0.2186	0.2564
	RND (T)	0.6591	0.6905	0.7135	0.3274	0.3434
	SAFE	0.7197	0.5901	0.6634	0.1370	0.2684
	STAC	0.5833	0.7857	0.7292	0.0917	0.1711
	DART (Short)	**0.9924**	0.8086	**0.9040**	0.1896	0.2602
	DART (Long)	0.8636	**0.8769**	0.8842	0.0853	**0.1581**
	DART (Full)	0.9773	0.8165	0.9023	**0.0721**	0.1876

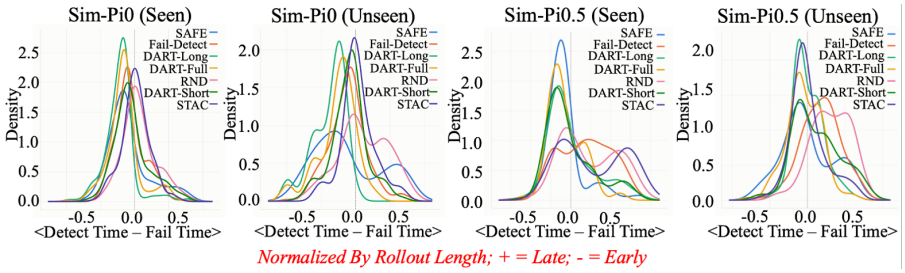


Figure 4: Detection timing error distributions in simulation.

accuracy using **Recall**, **Precision**, and **Balanced Accuracy (Bal. Acc.)** $= (\text{TPR} + \text{TNR})/2$. Temporal responsiveness is evaluated over detected failure trajectories using **Delay Time (DT)** $= \frac{1}{N_{\text{det}}} \sum_i \max(0, e_i)$ and **Absolute Timing Error (AT)** $= \frac{1}{N_{\text{det}}} \sum_i |e_i|$.

4.2 Comparison with Baselines

Table 1 presents the overall comparison across different VLA policies and task splits. Across all settings, DART maintains strong Recall, Precision, and Balanced Accuracy (Bal. Acc.) while substantially improving temporal responsiveness, with the advantage particularly evident on unseen tasks. On π_0 unseen tasks, Short achieves the highest Bal. Acc., while Long achieves the lowest DT, highlighting the distinct strengths of the two horizons. Although its AT is not always the lowest, the near-zero DT together with the timing distributions indicates that much of the temporal deviation arises from anticipatory rather than delayed alarms. On $\pi_{0.5}$ unseen tasks, DART achieves substantially higher detection accuracy than the baselines while retaining strong temporal responsiveness. Across

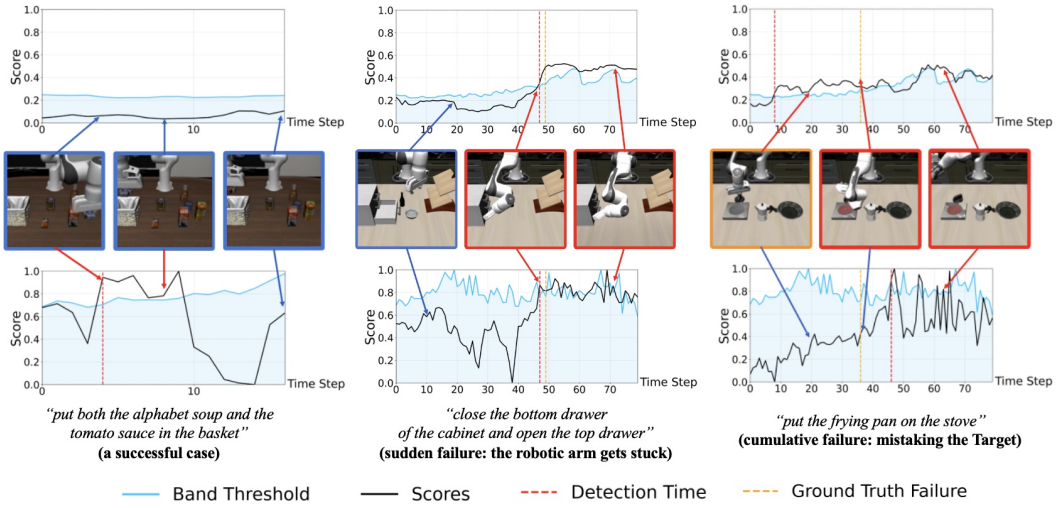


Figure 5: Qualitative comparison between DART (upper row) and Fail-Detect (lower row). *Left*: During successful execution, DART maintains relatively stable failure scores, whereas Fail-Detect exhibits larger score fluctuations under OOD execution. *Middle*: For a sudden failure, DART produces a temporally localized response near the failure manifestation time, while Fail-Detect exhibits larger score variations. *Right*: For a cumulative failure, DART produces an anticipatory response before the failure explicitly manifests, whereas Fail-Detect triggers after the manifestation time.

the variants, Short generally provides stronger detection accuracy, whereas Long tends to produce earlier alarms with lower DT; their fusion provides a favorable accuracy–timing trade-off across different policies and task distributions. Equally importantly, Fig. 4 further reveals temporal behaviors not fully captured by aggregate metrics. Across both π_0 and $\pi_{0.5}$, DART's timing distributions are generally concentrated around or slightly before the failure manifestation time, indicating consistent proactive detection, whereas several baselines exhibit distributions shifted toward positive temporal errors. In particular, despite competitive detection accuracy in some settings, Fail-Detect shows a stronger tendency toward delayed detection. We further provide qualitative comparisons across successful execution, sudden failure, and cumulative failure in Fig. 5.

4.3 Analysis of Dual-Horizon Failure Detection

To investigate the complementary roles of the two detection horizons, we evaluate DART (Short), DART (Long), and DART (Full) on sudden and cumulative failures in the π_0-unseen setting. As shown in Table 2, for sudden failures, Short achieves a substantially higher detection rate than Long (88.76% vs. 58.43%) and the lowest AT of 0.2079, suggesting that VLA internal semantic and control representations provide strong signals for abrupt execution inconsistencies. In contrast, for cumulative failures, Long exhibits stronger anticipatory behavior with a DT of 0.0000; its larger AT (0.4329) therefore mainly reflects early-detection bias rather than delayed detection. Fusing both horizons further increases the detection rate on cumulative failures to 91.45%, compared with 88.03% for Short and 85.47% for Long. These results demonstrate their complementary roles: Short is particularly effective for abrupt failures through immediate semantic and control cues, whereas Long provides anticipatory information for progressively developing failures, with fusion improving coverage across heterogeneous failure dynamics.

4.4 Effect of Horizon-Aligned Supervision

We compare current-step and future-shifted supervision for the Short and Long Horizon modules on π_0 unseen tasks. As shown in Table 3, Short performs better with current-step supervision, improving Bal. Acc. from 80.4% to 85.1% and reducing DT from 0.0853 to 0.0527. In contrast, Long benefits from future-shifted supervision, improving Bal. Acc. from 78.3% to 81.7% while maintaining near-zero DT. These results show that current-step labels favor immediate failure perception, whereas future-shifted labels better support failure anticipation.

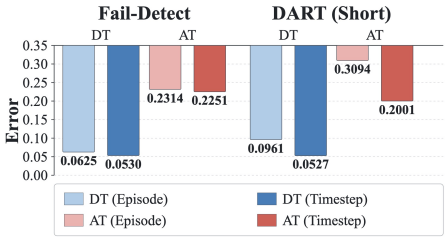
Table 2: Failure-type-specific detection performance on π_0-unseen. Detection Rate is the proportion of failures detected within per failure type. Timing metrics are computed over detected failures only.

Method	Det. Rate ↑	DT ↓	AT ↓
Sudden Failures ($N = 89$)			
DART (Short)	**0.8876**	0.0749	**0.2079**
DART (Long)	0.5843	**0.0005**	0.2877
DART (Full)	0.7640	0.0171	0.2164
Cumulative Failures ($N = 117$)			
DART (Short)	0.8803	0.0356	**0.1940**
DART (Long)	0.8547	**0.0000**	0.4329
DART (Full)	**0.9145**	0.0096	0.3329

Table 3: Comparison of failure detection performance between shifted and not shifted labels.

Horizon	Supervision	DT ↓	Bal. Acc. (%) ↑
Short	Current-Step	**0.0527**	**85.1**
	Future-Shifted	0.0853	80.4
Long	Current-Step	0.0025	78.3
	Future-Shifted	**0.0002**	**81.7**

Figure 6: Effect of timestep-level supervision on temporal responsiveness.



4.5 Effect of Timestep-Level Supervision

We compare Fail-Detect and DART (Short) under episode-level and timestep-level supervision on π_0 unseen tasks. As shown in Fig. 6, timestep-level supervision reduces both DT and AT for both methods. DART (Short) exhibits larger improvements than Fail-Detect, with DT decreasing from 0.0961 to 0.0527 and AT from 0.3094 to 0.2001. These results demonstrate the benefit of fine-grained supervision, with the larger gain of DART (Short) further suggesting its effectiveness in exploiting VLA internal semantic and control representations.

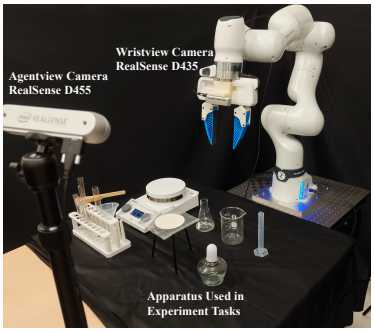


Figure 8: Robot and object settings in the real-world experiments

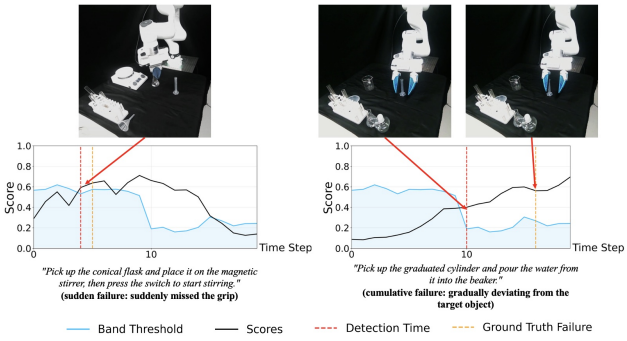


Figure 9: Comprehensive identification of real-world sudden failures and cumulative failures

4.6 Real-World Evaluation

We further evaluate DART on real-world manipulation trajectories, with quantitative results reported in Table 4. DART (Short) achieves the highest Bal. Acc. of 89.1% with 100% Recall and 88.4% Precision, while reducing DT to 0.0566 from 0.0800 for Fail-Detect. DART (Long) achieves the lowest DT of 0.0286, demonstrating stronger anticipatory behavior. DART (Full) maintains a high Bal. Acc. of 87.8% and Recall of 97.4%, while achieving the lowest AT of 0.1257. As shown in Fig. 7, DART's timing distributions remain concentrated around and slightly before failure manifestation, whereas the baselines are broader. This further confirms the complementary behavior of the two horizons: Short provides reliable failure perception, Long provides earlier anticipation, and their fusion balances the two. Representative sudden and cumulative failure cases are shown in Fig. 9.

Table 4: Quantitative comparison of failure detection performance in real-world experiments.

Method	Rec. ↑	Prec. ↑	B.Acc. ↑	DT ↓	AT ↓
Fail-Detect (T)	0.7895	0.8824	0.8078	0.0800	0.1667
RND (T)	**1.0000**	0.6230	0.5000	—	—
SAFE	0.7368	0.8485	0.7597	0.0696	0.2339
STAC	0.3235	0.7857	0.4475	0.0591	0.2045
DART (Short)	**1.0000**	**0.8837**	**0.8913**	0.0566	0.1342
DART (Long)	0.9211	0.7955	0.7649	**0.0286**	0.1900
DART (Full)	0.9737	0.8810	0.8781	0.0432	**0.1257**

Figure 7: Real-world detection timing error distributions.

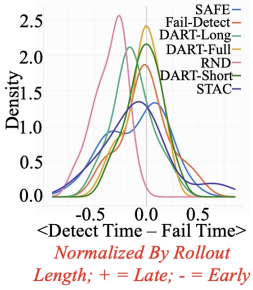
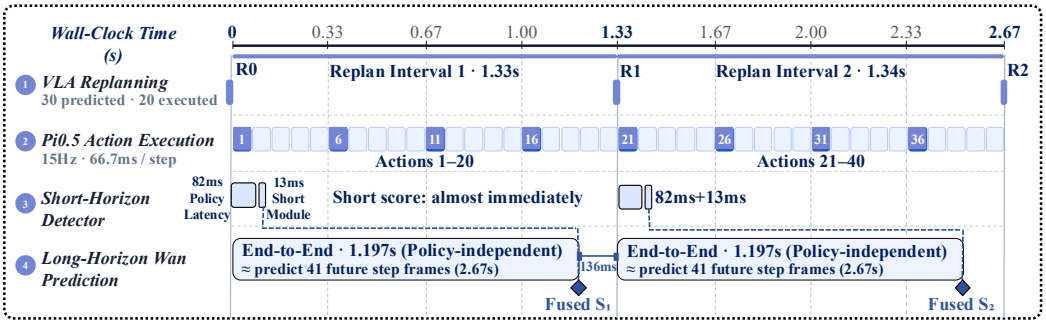


Figure 10: Runtime efficiency of DART in the real-world execution.



4.7 Runtime Efficiency Assessment

To further assess the practical feasibility of DART for real-world deployment, we analyze its wall-clock efficiency in real-world execution, as shown in Fig. 10. The $\pi_{0.5}$ policy operates at 15 Hz and replans every 20 executed actions, corresponding to a 1.33 s interval. At each replanning step, $\pi_{0.5}$ policy inference takes approximately 82 ms, followed by only 13 ms of additional Short-Horizon scoring, while the Long-Horizon Wan inference takes 1.197 s on a single H100 GPU and predicts about 2.67 s of future evolution. Since Long-Horizon inference runs in parallel with action execution and completes before the next replanning step, DART introduces limited runtime overhead without blocking the VLA control loop.

5 Conclusion

We presented DART, a dual-horizon framework for proactive runtime failure detection in VLA policies. DART combines VLA internal semantic and control representations for short-horizon failure perception with world-model-predicted latents for long-horizon failure anticipation. Together with VLM-generated timestep-level supervision, DART achieves a favorable accuracy–timing trade-off across simulation and real-world experiments, demonstrating the complementary roles of the two detection horizons.

6 Limitations

DART has several limitations. First, the Short-Horizon module relies on architecture-specific VLA internal representations, which may limit transfer across heterogeneous policy backbones. Second, the Long-Horizon module requires environment-specific fine-tuning of the video world model, introducing additional adaptation cost and limiting scalability under large domain shifts. Finally, VLM-generated timestep-level supervision may still be affected by visual ambiguity and labeling noise. Future work will explore more transferable representations, generalizable world models, and more robust automated supervision.

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*, Proceedings of Machine Learning Research, pp. 689–723. PMLR, 2024. URL https://proceedings.mlr.press/v270/agia25a.html.

I. Apanasevich, M. Artemyev, R. Babakyan, P. Fedotova, Denis Grankin, E. Kupryashin, A. Misailidi, D. Nerus, A. Nutalapati, G. Sidorov, I. Efremov, M. Gerasyov, D. Pikurov, Y. Senchenko, Sergei Davidenko, D. Kulikov, M. Sultankin, K. Askarbek, O. Shamanin, Dmitrii Statovoy, E. Zalyaev, I. Zorin, A. Letkin, E. Rusakov, A. Silchenko, V. Vorobyov, S. Sobolnikov, and Aleksey Postnikov. Green-vla: Staged vision-language-action model for generalist robots. *CoRR*, abs/2602.00919, 2026. doi: 10.48550/ARXIV.2602.00919. URL https://doi.org/10.48550/arXiv.2602.00919.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey A. Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bosnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier J. Hénaff, Xi Xiong, Radu Soricut, Jeremiah Harmsen, and Xiaohua Zhai. Paligemma: A versatile 3b VLM for transfer. *CoRR*, abs/2407.07726, 2024. doi: 10.48550/ARXIV.2407.07726. URL https://doi.org/10.48550/arXiv.2407.07726.

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.

Jake Bruce, Michael D. Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal M. P. Behbahani, Stephanie C. Y. Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott E. Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. 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*, Proceedings of Machine Learning Research, pp. 4603–4623. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/bruce24a.html.

Jiafei Duan, Wilbert Pumacay, Nishanth Kumar, Yi Ru Wang, Shulin Tian, Wentao Yuan, Ranjay Krishna, Dieter Fox, Ajay Mandlekar, and Yijie Guo. AHA: A vision-language-model for detecting and reasoning over failures in robotic manipulation. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=JVkdSi7Ekg.

Milan Ganai, Rohan Sinha, Christopher Agia, Daniel Morton, and Marco Pavone. Real-time out-of-distribution failure prevention via multi-modal reasoning. *CoRR*, abs/2505.10547, 2025. doi: 10.48550/ARXIV.2505.10547. URL https://doi.org/10.48550/arXiv.2505.10547.

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.

Clémence Grislain, Hamed Rahimi, Olivier Sigaud, and Mohamed Chetouani. I-failsense: Towards general robotic failure detection with vision-language models. *CoRR*, abs/2509.16072, 2025. doi: 10.48550/ARXIV.2509.16072. URL https://doi.org/10.48550/arXiv.2509.16072.

Chenyang Gu, Jiaming Liu, Hao Chen, Runzhong Huang, Qingpo Wuwu, Zhuoyang Liu, Xiaoqi Li, Ying Li, Renrui Zhang, Peng Jia, Pheng-Ann Heng, and Shanghang Zhang. Manualvla: A unified VLA model for chain-of-thought manual generation and robotic manipulation. *CoRR*, abs/2512.02013, 2025a. doi: 10.48550/ARXIV.2512.02013. URL https://doi.org/10.48550/arXiv.2512.02013.

Qiao Gu, Yuanliang Ju, Shengxiang Sun, Igor Gilitschenski, Haruki Nishimura, Masha Itkina, and Florian Shkurti. SAFE: multitask failure detection for vision-language-action models. *CoRR*, abs/2506.09937, 2025b. doi: 10.48550/ARXIV.2506.09937. URL https://doi.org/10.48550/arXiv.2506.09937.

Danijar Hafner, Timothy P. Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, Proceedings of Machine Learning Research, pp. 2555–2565. PMLR, 2019. URL http://proceedings.mlr.press/v97/hafner19a.html.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy P. Lillicrap. Mastering diverse domains through world models. *CoRR*, abs/2301.04104, 2023. doi: 10.48550/ARXIV.2301.04104. URL https://doi.org/10.48550/arXiv.2301.04104.

Nantian He, Shaohui Li, Zhi Li, Yu Liu, and You He. Rediffuser: Reliable decision-making using a diffuser with confidence estimation. In *International Conference on Machine Learning (ICML)*, 2024.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Ben Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alex Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi^*_{0.6}$: a VLA that learns from experience. *CoRR*, abs/2511.14759, 2025a. doi: 10.48550/ARXIV.2511.14759. URL https://doi.org/10.48550/arXiv.2511.14759.

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, 2025b. doi: 10.48550/ARXIV.2504.16054. URL https://doi.org/10.48550/arXiv.2504.16054.

Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, Vedant Choudhary, Foster Collins, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Maitrayee Dhaka, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachlan Groom, Haroun Habeeb, Hunter Hancock, Karol Hausman, Gashon Hussein, Victor Hwang, Brian Ichter, Connor Jacobsen, Szymon Jakubczak, Rowan Jen, Tim Jones, Gregg Kammerer, Ben Katz, Liyiming Ke, Mairbek Khadikov, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Brendon LeCount, Sergey Levine, Xinyu Li, Adrian Li-Bell, Vladislav

Lialin, Zhonglin Liang, Wallace Lim, Yao Lu, Enyu Luo, Vishnu Mano, Nandan Marwaha, Aikys Mongush, Liam Murphy, Suraj Nair, Tyler Patterson, Karl Pertsch, Allen Z. Ren, Gavin Schelske, Charvi Sharma, Baifeng Shi, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Jiaming Tang, Jimmy Tanner, Shalom Tekeste, Marcel Torne, Kyle Vedder, Quan Vuong, Anna Walling, Haohuan Wang, Jason Wang, XuDong Wang, Chris Whalen, Samuel Whitmore, Blake Williams, Charles Xu, Sukwon Yoo, Lili Yu, Wuming Zhang, Zhuoyang Zhang, and Ury Zhilinsky. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities. *CoRR*, abs/2604.15483, 2026. doi: 10.48550/ARXIV.2604.15483. URL https://doi.org/10.48550/arXiv.2604.15483.

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*, Proceedings of Machine Learning Research, pp. 2679–2713. PMLR, 2024. URL https://proceedings.mlr.press/v270/kim25c.html.

James Law. Review of "algorithmic learning in a random world by vovk, gammerman and shafer", springer, 2005, ISBN: 0-387-00152-2. *SIGACT News*, 37(4):38–40, 2006. doi: 10.1145/1189056.1189065. URL https://doi.org/10.1145/1189056.1189065.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

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, 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*, Proceedings of Machine Learning Research, pp. 4286–4313. PMLR, 2024. URL https://proceedings.mlr.press/v270/liu25g.html.

Qiang Liu. Rectified flow: A marginal preserving approach to optimal transport. *CoRR*, abs/2209.14577, 2022. doi: 10.48550/ARXIV.2209.14577. URL https://doi.org/10.48550/arXiv.2209.14577.

Yecheng Jason Ma, Joey Hejna, Chuyuan Fu, Dhruv Shah, Jacky Liang, Zhuo Xu, Sean Kirmani, Peng Xu, Danny Driess, Ted Xiao, Osbert Bastani, Dinesh Jayaraman, Wenhao Yu, Tingnan Zhang, Dorsa Sadigh, and Fei Xia. Vision language models are in-context value learners. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=friHAl5ofG.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: efficient action tokenization for vision-language-action models. *CoRR*, abs/2501.09747, 2025. doi: 10.48550/ARXIV.2501.09747. URL https://doi.org/10.48550/arXiv.2501.09747.

Yiran Qin, Zhelun Shi, Jiwen Yu, Xijun Wang, Enshen Zhou, Lijun Li, Zhenfei Yin, Xihui Liu, Lu Sheng, Jing Shao, Lei Bai, and Ruimao Zhang. Worldsimbench: Towards video generation models as world simulators. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, Proceedings of Machine Learning Research. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/qin25f.html.

Ralf Römer, Adrian Kobras, Luca Worbis, and Angela P. Schoellig. Failure prediction at runtime for generative robot policies. *CoRR*, abs/2510.09459, 2025. doi: 10.48550/ARXIV.2510.09459. URL https://doi.org/10.48550/arXiv.2510.09459.

Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *J. Mach. Learn. Res.*, 9: 371–421, 2008. doi: 10.5555/1390681.1390693. URL https://dl.acm.org/doi/10.5555/1390681.1390693.

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.

Qwen Team. Qwen3-vl technical report. *CoRR*, abs/2511.21631, 2025. doi: 10.48550/ARXIV.2511.21631. URL https://doi.org/10.48550/arXiv.2511.21631.

Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Xiaofeng Meng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *CoRR*, abs/2503.20314, 2025. doi: 10.48550/ARXIV.2503.20314. URL https://doi.org/10.48550/arXiv.2503.20314.

Yilin Wu, Ran Tian, Gokul Swamy, and Andrea Bajcsy. From foresight to forethought: Vlm-in-the-loop policy steering via latent alignment. *CoRR*, abs/2502.01828, 2025. doi: 10.48550/ARXIV.2502.01828. URL https://doi.org/10.48550/arXiv.2502.01828.

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.

Shijie Zhou, Bin Zhu, Jiarui Yang, Xiangyu Zhao, Jingjing Chen, and Yu-Gang Jiang. RC-NF: robot-conditioned normalizing flow for real-time anomaly detection in robotic manipulation. *CoRR*, abs/2603.11106, 2026. doi: 10.48550/ARXIV.2603.11106. URL https://doi.org/10.48550/arXiv.2603.11106.

A Appendix

This supplementary material provides additional implementation and evaluation details for DART. We first describe the simulation and real-world evaluation protocols, followed by the human annotation procedure and an analysis of the consistency between VLM-generated timestep-level supervision and human annotations. We then provide additional implementation details of the Long-Horizon module and the hyperparameter configurations. Finally, we present the prompt design used for automated timestep-level labeling and additional qualitative results from the real-world experiments.

A.1 Simulation Evaluation Details

We evaluate DART in the LIBERO benchmark using trajectories collected from two VLA policies, π_0 and $\pi_{0.5}$. The evaluation covers diverse manipulation tasks with different spatial configurations, goals, objects, and long-horizon interaction patterns. To evaluate generalization beyond the training task distribution, the collected tasks are divided into seen and unseen subsets, where detector training and calibration are performed using trajectories from seen tasks, while unseen tasks are reserved for evaluating task-level generalization.

During environment execution, both VLA policies periodically replan actions from the latest observation and language instruction. For π_0, policy replanning is performed every 5 timesteps with an action prediction horizon of 50 steps. For $\pi_{0.5}$, replanning is also performed every 5 timesteps, with an action prediction horizon of 10 steps. At each replanning step, we record the observations, predicted actions, and internal VLA representations required by the Short-Horizon module. Specifically, we extract the Vision-Language (VL) states from the multimodal backbone and the Hidden Control State from the action expert, following the feature extraction procedure described in the main paper.

Upon completion of each rollout, the full execution video and corresponding policy information are stored for offline processing. The proposed VLM-based timestep-level labeling pipeline is then applied to the recorded trajectories to generate dense timestep-level supervision. Human annotations are additionally collected for failed evaluation trajectories to determine the failure manifestation time used for computing temporal evaluation metrics. Importantly, the VLM-generated timestep-level labels are used for detector training, whereas the human-annotated manifestation times serve as the reference points for evaluation.

For failure detector training and calibration, trajectories from seen tasks are separated into training and calibration subsets. The training subset is used to optimize the failure scoring networks, while successful trajectories in the calibration subset are used to determine the detection threshold through conformal prediction. The unseen task split is reserved for evaluating the task-level generalization of the failure detector.

A.2 Real-World Evaluation Details

To evaluate DART beyond simulation, we conduct physical robot experiments on laboratory manipulation tasks that require precise object interaction and multi-stage execution. The experiments are performed using a Franka robot equipped with an Agentview RealSense D455 camera and a Wristview RealSense D435 camera, providing both global scene observations and local visual observations around the end effector.

We consider four representative laboratory manipulation tasks, as illustrated in Fig. 11:

1. **Beaker Placement**: Pick up the beaker and place it on the alcohol lamp.
2. **Test Tube Heating**: Pick up the test tube with a clamp and hold it over the alcohol lamp for heating.
3. **Magnetic Stirring**: Pick up the conical flask, place it on the magnetic stirrer, and press the switch to start stirring.
4. **Liquid Pouring**: Pick up the graduated cylinder and pour the water into the beaker.

Physical executions are controlled by a $\pi_{0.5}$ policy fine-tuned on our laboratory manipulation dataset. During deployment, the policy replans every 20 timesteps and predicts an action horizon of 30 future steps at each replanning step. The same VLA internal representations used in simulation are recorded during execution for Short-Horizon failure scoring, while visual observations and task instructions provide the conditioning information required by the Long-Horizon module.

The real-world trajectories contain both successful executions and failures arising from diverse execution dynamics, including abrupt control or interaction errors and progressively accumulated deviations. Each rollout is fully recorded from the robot cameras, and the corresponding policy information is synchronized with the visual observations. The same VLM-based annotation pipeline used in simulation is applied offline to generate timestep-level supervision, ensuring a consistent labeling procedure across simulation and physical experiments.

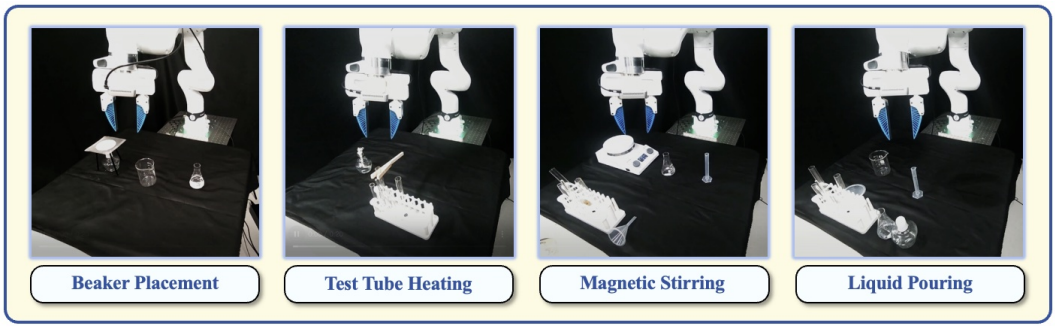


Figure 11: Real-world manipulation tasks used for physical robot evaluation.

For evaluation, human annotators additionally identify the failure manifestation time for each failed trajectory according to the annotation protocol described in Sec. A.3. This provides a common temporal reference for computing DT and AT in both simulation and real-world experiments.

A.3 Human Annotation Protocol

Human annotations are used only to establish the reference failure manifestation time for evaluation. For each failed trajectory, we define the **failure manifestation time** t_{fail} as the earliest timestep at which the execution failure becomes explicitly and unambiguously observable from the robot state, action, and resulting environment state. Importantly, this definition corresponds to the observable manifestation of a failure rather than the preceding buildup of potentially risky behavior.

For example, when an object gradually tilts before falling, the preceding tilt is regarded as a developing risk, while the timestep at which the object actually falls and the execution failure becomes unambiguous is annotated as t_{fail}. Similarly, transient deviations or early warning signs that remain recoverable are not treated as failure manifestation; the annotation is placed at the earliest point where the failure itself becomes explicitly and unambiguously observable. This definition allows detections before t_{fail} to be interpreted as proactive alarms rather than annotation errors.

All failed simulation and real-world trajectories used for temporal evaluation are annotated following the same protocol. Each trajectory is first labeled by one annotator and subsequently re-labeled by a second annotator, with disagreements resolved through re-examination of the corresponding execution sequence.

We further distinguish the human evaluation annotation from the timestep-level labels used for detector training. The human annotation provides a single reference manifestation time for each failed trajectory, whereas the VLM-based labeling pipeline generates dense timestep-level failure scores for supervision. In the VLM labeling process, a timestep may be assigned an elevated failure-related score even if the policy subsequently recovers, since the purpose of these labels is to capture emerging execution risks rather than only terminal failures.

A.4 Consistency Between VLM Supervision and Human Annotations

We further examine whether the automatically generated VLM supervision is temporally consistent with human annotations. This analysis is conducted on failed π_0-LIBERO trajectories for which human-annotated failure manifestation times are available.

For analysis only, we identify the first timestep at which the dense VLM-generated score A_t exceeds a threshold of 0.5:

$$t_{\text{VLM}} = \min\{t \mid A_t \geq 0.5\}. \tag{1}$$

This threshold crossing is not used as the evaluation ground truth; it only provides a diagnostic measure of the temporal consistency between the automatically generated supervision and human annotations. Following the temporal normalization used in the main paper, for trajectories in which such a threshold crossing exists, we compute

$$e_{\text{VLM}} = \frac{t_{\text{VLM}} - t_{\text{fail}}}{T}, \tag{2}$$

where t_{fail} is the human-annotated failure manifestation time and T denotes the rollout length. Negative values indicate that the VLM score crosses the threshold before the human-annotated failure manifestation time, while positive values indicate a later crossing.

Using the 0.5 threshold, the VLM produces a valid threshold crossing for approximately 86% of the evaluated failed trajectories. As shown in Fig. 12, the temporal-error distribution is concentrated around zero, indicating that the automatically generated supervision is generally temporally aligned with human annotations. The distribution also exhibits a mild tendency toward negative errors, which is consistent with the intended role of the VLM pipeline: elevated failure scores may arise from emerging abnormal behavior before the failure becomes explicitly observable according to the human manifestation criterion.

Importantly, the VLM-generated labels are not used as evaluation ground truth. Human annotations define the reference failure manifestation time for evaluation, whereas the VLM pipeline provides scalable and dense timestep-level supervision for training the failure detector.

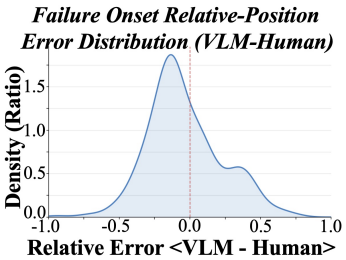


Figure 12: Distribution of the normalized temporal difference between the VLM score threshold crossing and the human-annotated failure manifestation time on failed π_0-LIBERO trajectories. Negative values correspond to earlier VLM responses and positive values correspond to later responses.

A.5 Additional Long-Horizon Implementation Details

To adapt the pre-trained Wan video generation model for future state prediction in robotic manipulation, we fine-tune it using execution sequences collected from the corresponding manipulation environments. In addition to the current visual observation and task instruction, temporal progress is introduced as conditioning information.

For a trajectory of length T, the normalized execution progress at timestep t is defined as

$$\tau_t = \frac{t}{T} \in [0, 1]. \tag{3}$$

The conditioning context is constructed as

$$\mathcal{C} = [\mathbf{x}_0 \parallel \tau_t \parallel \mathbf{c}], \tag{4}$$

where $\mathbf{x}_0$ denotes the visual observation, $\mathbf{c}$ denotes the task instruction, and $\parallel$ represents conditioning concatenation. Incorporating execution progress helps the model generate future latent representations that are temporally aligned with the current stage of task execution.

During fine-tuning, the decoder provides pixel-space reconstruction supervision to keep the predicted latent dynamics grounded in plausible future observations. During failure detection inference, however, the decoder is not required: DART directly uses the predicted latent representations produced by the world model for Long-Horizon failure scoring.

A.6 Implementation Details and Hyperparameter Configurations

The Short-Horizon branch processes the VLA's internal semantic and control representations using lightweight scoring networks, while the Long-Horizon branch maps the predicted Wan latent representation to a failure score. The principal training and architectural hyperparameters are summarized in Table 5.

For the semantic representation, each base-camera, wrist-camera, and language feature has dimension 2048. Their concatenation forms the 6144-dimensional Vision-Language representation described in the main paper. The Hidden Control State extracted from the action expert has dimension 1024.

Table 5: Comprehensive hyperparameter configurations of the failure detection network.

Category	Hyperparameter	Value
Data & Windows	Batch Size (`batch_size`)	128
	Hidden State Windows (`hid_steps`)	4
	Vision-Language Windows (`vl_steps`)	4
Input Dimension	Hidden State Dimension (`input_hid_dim`)	1024
	Per-stream VL Feature Dimension	2048
	Concatenated VL State Dimension	6144
Network Architecture	Internal Hidden Dimension (`hidden_dim`)	256
	Dropout Rate (`dropout`)	0.1
	Final Activation Layer (`final_act_layer`)	`sigmoid`
Modular Configuration	Hidden Model Type (`hid_model`)	`mlp`
	Vision-Language Model Type (`vl_model`)	`mlp`
	Wan Predictive Model Type (`wan_model`)	`mlp`
Optimization & Loss	Total Training Epochs (`n_epochs`)	100
	Initial Learning Rate (`lr`)	3×10^{-5}
	LR Step Size (`lr_step_size`)	300
	LR Decay Gamma (`lr_gamma`)	1

A.7 Prompt Design

To generate dense timestep-level supervision from execution videos, we use a structured prompting strategy that combines explicit temporal context, failure-mode decomposition, and standardized output constraints. The prompt design consists of three components:

- **Contextual Injection and Feature Extraction.** The prompt specifies that the input video corresponds to a local temporal segment of a complete manipulation trajectory. The VLM is first instructed to describe the observable execution process, including robot motion, end-effector behavior, object displacement, and task progress, before producing a failure judgment. This encourages the prediction to be grounded in observable visual evidence rather than relying only on a final classification response.
- **Fine-Grained Failure Taxonomy.** We divide execution states into five categories that cover representative failure patterns: (1) *Success*, corresponding to normal and continuous task execution; (2) *Physical Perturbation*, such as objects falling or being overturned; (3) *Control Instability*, including shaking, prolonged stationarity, or unstable robot motion; (4) *Semantic Mismatch*, where the executed behavior is inconsistent with the task instruction; and (5) *Interaction Failure*, such as missing the target object during grasping or manipulation.
- **Standardized Output Constraints.** To enable automatic large-scale processing, the VLM is required to return its final prediction using a fixed output format. Specifically, the final sentence follows the form "Final classification result is i", allowing the predicted category to be directly parsed for subsequent label generation.

The five-way categorical prediction is first mapped to a binary normal/abnormal judgment at the segment level, where normal execution is treated as the non-failure state and the remaining abnormal categories are treated as failure-related states. Predictions from multiple temporal scales are then aggregated at each anchor point and temporally smoothed, yielding the continuous dense supervision A_t used for detector training.

A.8 Additional Real-World Qualitative Results

We provide additional qualitative comparisons of failure score trajectories under three representative real-world execution conditions: successful execution, sudden failure, and cumulative failure. These examples complement the real-world quantitative evaluation in the main paper and further illustrate the temporal behaviors of different failure detection methods.

As shown in Fig. 13, during successful execution, DART maintains relatively stable failure scores without triggering unnecessary alarms, whereas several baseline methods exhibit larger score fluctuations. For the sudden failure shown in Fig. 14, the robot exhibits an abrupt control error during execution, and DART responds closely to the failure manifestation time. Finally, Fig. 15 presents a cumulative failure in which small execution deviations progressively accumulate before the final failure becomes explicitly observable. In this case, DART produces anticipatory failure signals before complete execution breakdown.

These additional qualitative examples complement the quantitative analyses in the main paper by illustrating the temporal evolution of failure scores under successful executions, sudden failures, and cumulative failures.

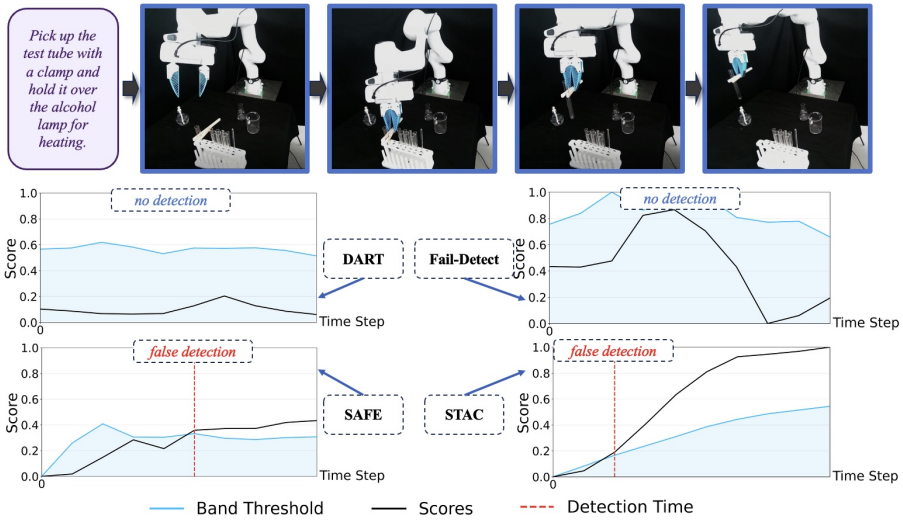


Figure 13: Comparison of failure detection scores on a successful real-world trajectory.

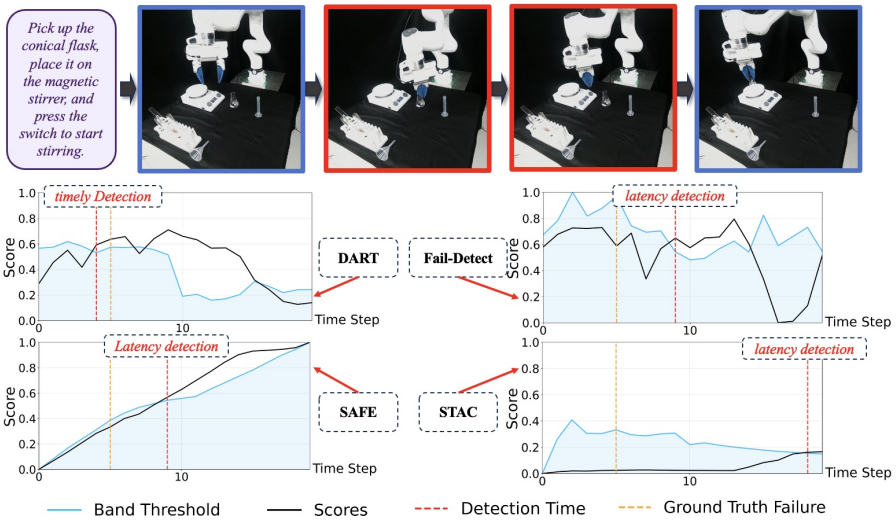


Figure 14: Comparison of failure detection methods on a real-world trajectory with a sudden failure.

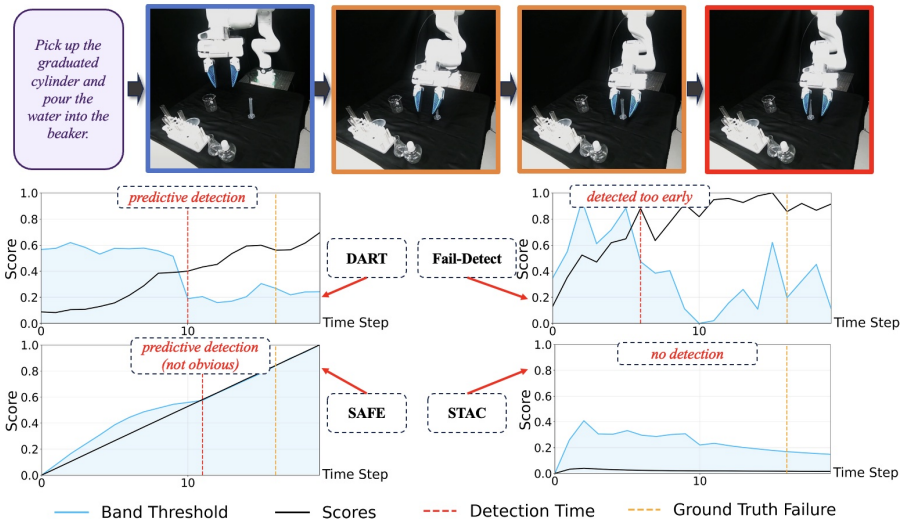


Figure 15: Comparison of failure detection methods on a real-world trajectory with a cumulative failure.