From Perception Logs to Failure Modes: Language-Driven Semantic Clustering of Failures for Robot Safety

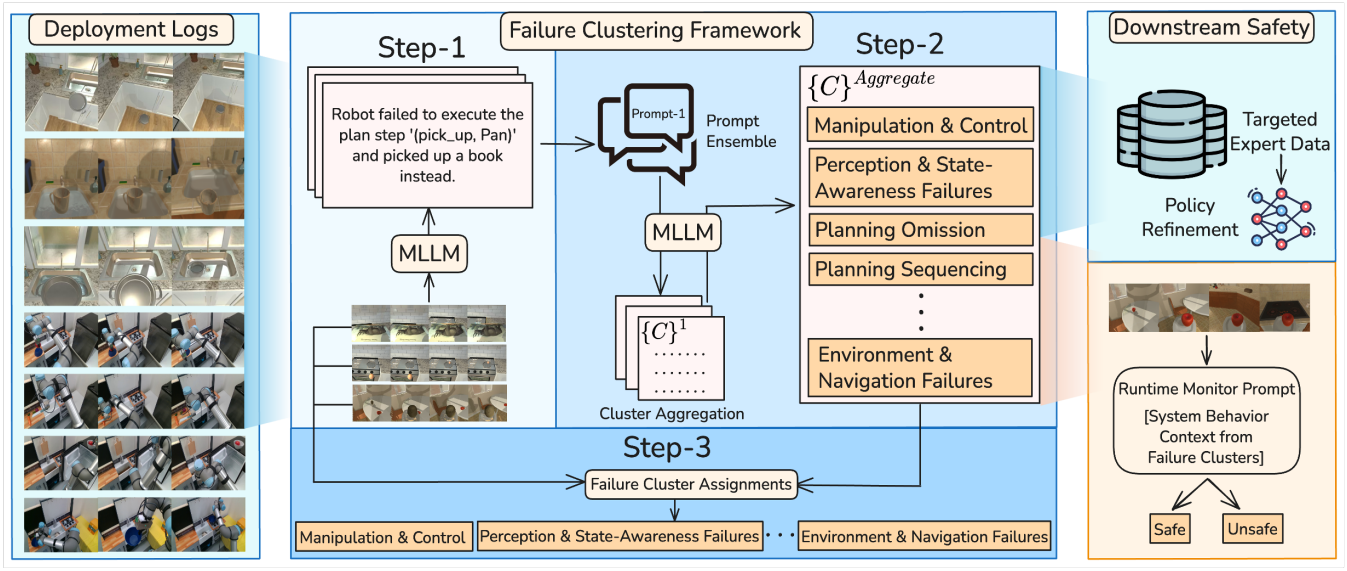

Fig. 1: A closed-loop, language-driven framework for interpretable failure mode discovery in autonomous systems. It extracts semantically meaningful failure patterns from deployment-time perceptual data without supervision and organizes them into human-understandable clusters. These clusters support downstream applications such as targeted data collection, policy refinement, and runtime failure monitoring, enabling scalable and continuous safety improvement.

***Abstract*— As robotic systems become increasingly integrated into real-world environments—ranging from autonomous vehicles to household assistants—they inevitably encounter diverse and unstructured scenarios that lead to failures. While such failures pose safety and reliability challenges, they also provide rich perceptual data for improving future performance. However, manually analyzing large-scale failure datasets is impractical. In this work, we present a method for automatically organizing large-scale robotic failure data into semantically meaningful failure clusters, enabling scalable learning from failure without human supervision. Our approach leverages the reasoning capabilities of Multimodal Large Language Models (MLLMs), trained on internet-scale data, to infer high-level failure causes from raw perceptual trajectories and discover interpretable structure within uncurated failure logs. These semantic clusters reveal patterns and hypothesized causes of failure, enabling scalable learning from experience. We demonstrate that the discovered failure modes can guide targeted data collection for policy refinement, accelerating iterative improvement in agent policies and overall safety. Additionally, we show that these semantic clusters can benefit online failure monitoring systems, offering a lightweight yet powerful safeguard for real-time operation. We demonstrate that this framework enhances robot learning and robustness by transforming real-world failures into actionable and interpretable signals for adaptation. Wesbite: https://mllm-failure-clustering.github.io/**

I. INTRODUCTION

Autonomous systems—ranging from self-driving vehicles to household robots—are increasingly deployed in open, dynamic environments. In such unstructured settings, even state-of-the-art robotic systems are prone to failures due to unexpected interactions, unmodeled dynamics, and long-tail edge cases that deviate from training distributions. Traditional validation pipelines, often grounded in simulation or controlled testing, struggle to capture the full complexity of real-world deployment, leaving many failure modes undetected until operation.

A promising direction for improving robustness is to systematically learn from failures that occur during deployment. Robots naturally collect large volumes of perceptual data, including traces of both successful and failed interactions. These failure trajectories can provide valuable insights about the underlying conditions that led to safety violations, brittleness, or policy errors. However, manually curating and analyzing large-scale failure data is time-consuming and fundamentally unscalable.

In this work, we propose a framework for automatically organizing failure trajectories into semantically meaningful clusters, enabling scalable, unsupervised learning from failures. Our method leverages the reasoning capabilities of MLLMs—pretrained on internet-scale vision-language data—to infer high-level failure causes directly from raw observations. By reasoning over sequences of perceptual inputs, we identify structure in uncurated failure logs, grouping them into interpretable categories described in natural language. Importantly, our framework operates in a completely unsupervised manner, obviating the need for costly human annotation while still isolating nontrivial keywords and cues associated with specific error causes.

The resulting semantic clusters bring multiple downstream benefits. In this paper, we demonstrate their value in guiding

targeted data collection, enabling developers to focus training efforts on critical or underrepresented failure scenarios, thereby reducing data collection costs. Additionally, these clusters can be integrated directly into online failure monitoring systems, providing a layer of semantic failure detection that acts as an early-warning mechanism for potential system safety violations at runtime. Beyond these applications, identifying semantic failure modes opens further avenues, such as facilitating targeted stress testing.

Our framework not only supports large-scale, structured analysis of failure data but also emphasizes interpretability – crucial for deployment in safety-critical domains. By revealing a structured understanding of failure causes in terms amenable to human interpretation, we offer actionable insights that can help stakeholders better assess the weaknesses of deployed systems and iterate more effectively.

Contributions. (a) we propose a novel framework that uses MLLMs to cluster robotic failure data into semantically meaningful groups; (b) our method infers failure causes directly from raw perceptual sequences, eliminating manual annotation or supervision; (c) we demonstrate the effectiveness of our approach on large-scale failure datasets, as well as showcase its potential for supporting downstream applications, including targeted data collection and online failure detection; (d) our framework emphasizes interpretability, generating natural language summaries and keywords for each failure mode to support human-in-the-loop diagnosis.

II. Related Works

Semantic Clustering of Images with LLMs. Recent work has shown the effectiveness of MLLMs for semantic image grouping. Prior methods cluster images using human-specified language criteria [1] or automatically discover clustering criteria from image collections [2], removing manual input dependency. Other approaches highlight semantic differences between image sets [3], or focus on identifying important subpopulations or selecting semantically diverse subsets for training [4],[5]. Our work builds on this line of research by leveraging MLLMs for semantic clustering; however, we focus specifically on failure trajectories in autonomous systems rather than static images, introducing a richer temporal and causal structure to the clustering task.

Text Clustering and Topic Modeling. Recent topic modeling methods can be broadly grouped into three categories. First, extensions of classical topic modeling techniques [6]–[9] like Latent Dirichlet Allocation (LDA) integrate word embeddings to enhance semantic representation [10]–[13]. Second, fully embedding-based approaches [14]–[17] that can leverage contextualized representations from pre-trained language models. Third, methods that separate clusters generation from topic representation, allowing for a flexible topic model [18]–[20]. BERTopic [20], extends this approach by incorporating a class-based variant of Term Frequency-Inverse Document Frequency (TF-IDF). Our work differs in the way we use semantic reasoning to discover failure clusters from multimodal data and language-based explanations.

Failure Mining in Autonomous Systems. Falsification has emerged as a prominent methodology for uncovering failures in autonomous systems. A variety of approaches [21]–[27] have been proposed, wherein systems are tested in simulated environments with varied conditions designed to provoke failures. Such controlled testing allows researchers to identify the specific environmental parameters responsible for system breakdowns. However, while these methods effectively reveal failure scenarios tied to controlled variables, they fall short in capturing semantic failure modes of the system that require nuanced interpretation, requiring a human to manually go through those failure scenarios to identify failure modes of the system. In contrast, our approach leverages readily collected failure data to automatically discover semantic failure modes, bypassing manual human inspection.

Language-based Failure Reasoning in Robotics. Recently, several studies have explored integrating LLMs into robotic systems to generate language-based explanations of failures. Prior work has shown that LLMs can enhance diagnostic capabilities in manipulation tasks by producing informative failure descriptions and even guiding planners to correct those failures [28]–[30]. Building on these advances, our work focuses on failure mode analysis, aiming to automatically cluster and interpret semantically rich textual descriptions from raw failure data—thereby enabling improved monitoring, interpretability, and targeted data collection.

III. Problem Formulation: Finding Failure Clusters from Perception Recordings

We consider the problem of discovering semantic failure clusters from perception data collected during robotic failures. These clusters provide interpretable structure over uncurated failure logs and can be used for downstream tasks.

Formally, we are given a dataset of N sequences, each consisting of K perceptual observations leading to a failure:

$$D = \{o^n_{1:K}\}_{n=1}^{N}$$

where $o^n_{1:K} = (o^n_1, o^n_2, \ldots, o^n_K)$ denotes the observation sequence for the n-th failure case. The final observation o^n_K corresponds to the failure event. For example, in autonomous driving, o^n_K might be an image showing a rear-end collision.

Our goal is to construct a system H that maps this dataset to a set of L semantic clusters:

$$H : D \mapsto \{C_l = (s_l, D_l)\}_{l=1}^{L},$$

where each cluster C_l is characterized by a natural language summary s_l and a subset of sequences $D_l \subset D$ that share a common failure mode. These clusters reflect high-level *failure themes* derived from raw observations. For instance, a cluster might be: C_l = `Rear-End Collisions: Insufficient Following Distance`, in which case each sequence in D_l corresponds to a failure where an autonomous car did not maintain a safe following distance from the vehicle in front.

This formulation enables unsupervised discovery of interpretable failure modes directly from perceptual logs, providing a scalable mechanism for structuring real-world failure

data and guiding robust and adaptive learning.

IV. Method

A. Discovering Failure Modes using MLLMs

Inferring the cause of failure in a robot trajectory is a complex task that requires understanding the robot's environment, the agent's actions, interactions with other agents, and their consequences. Doing so at scale across diverse failure episodes calls for automated systems capable of extracting and reasoning over semantic patterns in raw failure data. Our approach proceeds in three stages: (1) inferring failure reasons from perception sequences using an MLLM, (2) discovering semantic failure clusters via prompted reasoning over inferred causes, and (3) assigning each sequence to one of the discovered clusters, as shown in Fig. 1.

Observation Downsampling. To represent the failure event compactly, we downsample the tail of each sequence. Specifically, from each failure sequence $o^n_{1:K}$, we select the final T frames, sampled from $o_{K-T:K}$ at a reduced frame rate. This balances the need to understand the temporal context of the failure with the limitations of the MLLM's context window.

1) ***Step 1: Inferring Failure Reasons with MLLMs****:* Each downsampled sequence is fed to a reasoning MLLM along with a structured prompt. The prompt first asks the model to describe the scene and agent behavior along the trajectory, and then to infer a plausible cause of failure. We adopt a Chain-of-Thought (CoT) prompting strategy [31] to improve grounding and interpretability of the inferred reasons.

2) ***Step 2: Discovering Semantic Failure Clusters****:* To uncover structure across failures, we provide all N inferred failure reasons to a reasoning LLM and prompt it to group them into L distinct, interpretable semantic clusters. The number of clusters "L" is not fixed, but dynamically decided by the LLM based on dataset-specific characteristics, such that each cluster represents a unique failure mode, with minimal overlap between clusters and comprehensive coverage of the dataset. The model is instructed to ensure that clusters are mutually exclusive and collectively exhaustive, facilitating clear categorization and downstream analysis.

Each cluster C_l is annotated with: a natural language name s_l for the cluster, a short description of the failure type, keywords capturing representative situations in the cluster, and an estimated frequency of occurrence in the dataset. These annotations serve both interpretability and downstream usage in safety-critical settings.

Automated Prompt Ensemble and Cluster Aggregation Since LLM outputs are sensitive to prompt phrasing [32], we generate a prompt ensemble to improve robustness inspired by [33]. Given an initial prompt, an LLM proposes three other prompts using good prompting practices. We use this ensemble of prompts to infer multiple failure clustering results. The resulting clusterings are then merged by the same reasoning LLM, like an aggregation model, that consolidates overlapping clusters and unifies labels and descriptions into a single set of failure clusters $\{C\}^{Aggregate}$. This approach generates a more comprehensive set of failure clusters by capturing diverse interpretations of failure reasons and compiling them into one final set of clusters.

3) ***Step 3: Assigning Trajectories to Failure Clusters****:* Finally, each failure trajectory is assigned to one of the L discovered clusters by prompting an LLM with the cluster names s_l, keywords, and descriptions generated in the previous step, and asking it to map the trajectory description and corresponding failure reason to one of the clusters. Trajectories that do not match any cluster are flagged as outliers, which may indicate rare or complex failure modes.

B. Downstream Safety Enhancement

Discovered failure clusters support two key mechanisms for enhancing safety in autonomous systems.

Online Failure Detection. Clusters, along with their associated keywords and explanations, form the foundation for semantic understanding in our MLLM-based failure detection system. During execution, an MLLM-based monitor observes a recent history of visual inputs and reasons about them in the context of known failure modes. It leverages the structured, semantic knowledge captured in these clusters—each describing characteristic failure types, contexts, and explanations—to interpret unfolding behavior. When the current observations semantically align with a known failure mode, the monitor flags the situation as unsafe, enabling early anticipation of failures before they fully manifest. This semantic understanding supports more nuanced, accurate, and proactive failure detection, improving the overall safety and reliability of the system at runtime.

Targeted Data Collection and Policy Refinement. Semantic clusters enable *targeted* data augmentation by identifying failure scenarios that require additional supervision. This allows practitioners to collect expert demonstrations or counterfactual data in these high-risk regimes and retrain or finetune the policy accordingly. Prior work has also shown that specification-guided or failure-aware data collection improves robustness in previously unsafe scenarios [34]–[37].

V. Experiments

We evaluate our framework for discovering semantic failure clusters across three distinct robotic domains. The first case study examines robot manipulation failures in long-horizon kitchen tasks, the second analyzes real-world dashcam videos of car crashes, and the third focuses on autonomous vision-based navigation in indoor office environments. In each domain, our goal is to extract interpretable clusters that represent underlying failure modes from raw perceptual sequences and to demonstrate their usefulness for downstream tasks such as runtime failure monitoring and targeted data collection, when applicable.

We propose to use Gemini 2.5 Pro as the MLLM for understanding system behavior and inferring failure reason of a trajectory, and OpenAI o4-mini for semantic cluster discovery, trajectory-to-cluster assignment, online failure detection, and as judge for metrics computations. Our choice of LLMs is based on their reasoning abilities for particular tasks, after evaluating different open and closed-source models. Sec. V-A

discusses the results for discovering failure modes and Sec. V-B shows their usage for downstream safety enhancement.

A. Discovering Failure Modes using MLLMs

1) ***Case Study 1: Robot Manipulation****:* We evaluate our framework on RoboFail [28], a dataset of manipulation failures in long-horizon kitchen tasks. The dataset includes an expert-defined taxonomy of eight distinct manipulation failure modes, comprising 100 simulated failure videos across 10 kitchen tasks and 30 real-world videos (960×960 resolution at 1 fps). Each video provides detailed task context, including the failure timestamp, task description & success conditions, the symbolic action plan executed by the robot, and an expert-annotated failure reason.

a) Inferring Failure Reasons: While inferring failure reasons, we incorporate task context and robot's action plan alongside perception inputs. These additional cues help the model distinguish between failures caused by flawed planning and those arising from execution errors. Fig. 2 shows an example inference of a failure caused due to dropping a pot, including our prompt and LLM's response. Table I reports results for both simulation and real-world data. To assess the inferred failure reasons, we use two metrics: (a) average cosine similarity (CS) and (b) LLM-as-a-Judge (LLM%) evaluation that uses an LLM to check semantic similarity between the inferred and expert-annotated labels.

TABLE I: Failure reasoning performance of different MLLMs evaluated on simulation and real-world data.

Method	Simulation		Real-World	
	CS	LLM(%)	CS	LLM(%)
Video-LLaVA-7B	39.78	10.2	65.23	10.0
Qwen2.5-VL-7B	52.48	25.0	67.36	33.3
OpenAI o4-mini	55.80	47.0	59.87	30.0
Gemini 2.5 Pro	**61.13**	**73.0**	**62.81**	**56.7**

Among the evaluated models, Gemini 2.5 Pro achieves the best performance across simulation and real-world datasets, outperforming other proprietary and open-source models. Notably, it attains a high cosine similarity (CS) and LLM-based semantic agreement (LLM) scores, indicating stronger alignment with expert-labeled failure reasons. Though LlaVA and Qwen models obtain higher CS values for real-world data, their lower (LLM) scores indicate that the inferred failure reasons lack true semantic agreement with expert annotations despite having similar contextual embeddings.

We also compare against AHA-13B [29], a VLM fine-tuned specifically on a manipulation failure dataset. On the real-world data, AHA-13B achieves 47.1 (CS) and 46.5 (LLM%), both lower than our proposed reasoning-based approach. In our experiments we observe that while fine-tuned model can specialize in specific viewpoints, tasks, and environments, it struggles to generalize across diverse tasks or unseen settings. In contrast, our framework, which leverages general-purpose reasoning models, remains viewpoint-agnostic and adapts effectively across different domains without retraining on expert-collected datasets. Since our method operates offline, the inherent latency of reasoning models does not pose a limitation. This allows leveraging their strong semantic understanding to infer nuanced and contextually accurate failure reasons across diverse conditions.

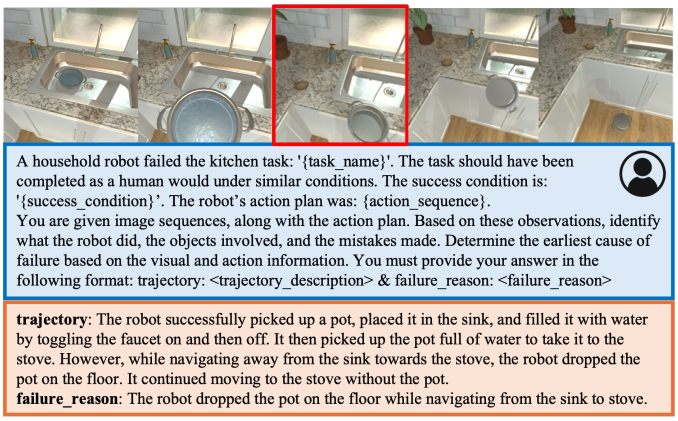

Fig. 2: A failure inference example where robot dropped a pot with water on the floor while carrying it. Frame with red border shows the frame at failure timestamp. Blue box shows the prompt and orange box shows LLMs response.

b) Discovering Semantic Clusters: We cluster all the obtained failure reasons into distinct semantic groups. The list below and Fig. 3 present the discovered clusters, their keywords, and occurrence frequencies in the dataset. Qualitatively, the clusters exhibit clear and interpretable groupings—for instance, `Manipulation & Control Failures` capture execution-level issues like failed grasps or unintended drops, `Planning Parameter & Resource Selection Errors` describe incorrect or misspecified actions, and `Perception & State-Awareness Issues` isolate failures due to misidentification or incomplete scene understanding. Together, these clusters capture fine-grained distinctions between perception, planning, and control errors, providing a semantically rich and interpretable organization of failures.

Robot Manipulation Clusters

(18%) Manipulation & Control Failures: drop, failed grasp, collision, put-in fail, poor pour, misplacement
(17%) Perception & State-Awareness Failures: misidentify, wrong object, object-state misperception, content-detection fail, occupancy detection
(17%) Planning Parameter & Resource Selection Errors: wrong container, inappropriate tool, unsafe object, incorrect burner, plan-specified target wrong
(16%) Planning Sequencing & Order Errors: wrong order, out-of-sequence, before/after swapped, precondition violation
(12%) Planning Logic & Efficiency Flaws: illogical, flawed assumption, inefficient loop, redundant step, violates domain rule
(10%) Planning Omission Errors: missing step, forgot to open/close, omit pick-up, no turn-on/off, neglect slice
(10%) Environment & Navigation Failures: path obstructed, blocked handle, surface occupied, wrong location, mispositioned

Fig. 3: Robot manipulation failure clusters with examples.

Baselines and Results. To evaluate the quality of the discovered failure clusters, we compare our approach against BERTopic [20], a state-of-the-art topic modeling method that combines transformer embeddings with unsupervised clustering and keyword extraction. We apply BERTopic to the same set of textual failure reasons produced in Step 1 to ensure a direct and fair comparison. We also include a stronger variant, BERTopic-LLM, in which a language model summarizes each cluster using representative keywords, summaries, and examples.

BERTopic Keywords

(41) the, water, to, mug, pot, it, robot, of, sink, with
(38) the, to, it, robot, egg, bowl, plan, pan, potato, is
(12) the, bread, slice, toaster, to, robot, failed, of, was, loaf
(9) the, remote, to, laptop, plan, on, it, television, navigate

BERTopic-LLM Clusters

(41) Robot planning failures
(38) Robot task failures
(12) Robot action errors
(9) Flawed robot plans

Standard BERTopic tends to produce clusters that lack semantic coherence, primarily grouping together frequently occurring words without capturing deeper conceptual relations. BERTopic-LLM improves interpretability by generating more meaningful topics, but its clusters remain broad, overlapping, and generic—often merging distinct categories such as `Wrong action order`, `Missing actions`, and `Wrong actions` into a single, high-level group like `Robot planning failures`. Consequently, it fails to capture root-cause distinctions critical for understanding robotic failure patterns.

For quantitative comparison, we compute LLM-generated similarity scores between the generated clusters and the expert-defined failure taxonomy from RoboFail by prompting an LLM to assign similarity scores for each generated cluster with each failure mode from the expert taxonomy. The heatmaps in Fig. 4 visualize these correspondences for our method (a) and baselines (b,c). Our approach yields sharper, diagonally dominant heatmaps—indicating strong one-to-one alignment between discovered and expert categories—while both BERTopic and BERTopic-LLM exhibit more diffuse and overlapping mappings. Moreover, our clusters achieve strong coverage across all eight expert-defined failure types, including both frequent and rare modes, demonstrating our framework's ability to recover the full taxonomy while maintaining semantic distinctness. This confirms that our LLM-based semantic clustering captures interpretable, non-redundant, and expert-consistent failure modes, outperforming topic-modeling baselines in both qualitative coherence and quantitative alignment.

c) Assigning Sequences to Clusters: Lastly, we assign all the textual failure reasons to the obtained list of clusters. Fig. 5 compares the weighted F-1 scores between the assignments made by our framework, the assignments made by an embedding-based assigner (EA)—which assigns to the cluster with the highest cosine similarity as the failure reason, and random assignment (RA).

Fig. 5: Weighted F-1 Score Comparison

Method	F-1 (%)
RA	13.93
EA	32.41
Ours	**85.53**

*2) **Case Study 2: Real-World Car Crash Videos**:* We apply our framework to the Nexar car crash dataset [38], which includes 1,500 dashcam videos (each approximately 40 seconds, 1280×720 resolution at 30 fps) of collisions or near-misses involving the ego vehicle. While these recordings are from human-driven vehicles, they serve as a proxy for autonomous driving failures in the absence of large-scale public AV failure datasets. However, our framework is directly applicable to autonomous vehicle logs as well.

Real-World Car Crash Clusters

(35%) Rear-End Collisions / Insufficient Following Distance: rear-end, tailgating
(25%) Intersection Right-of-Way Violations: left turn, red light
(18%) Unsafe Cut-In / Lane-Change Intrusions: lane change, cut-in
(8%) Lane Departure & Lateral-Clearance Errors: lane departure, misjudged gap
(7%) Visibility-Impaired Perception Failures: glare, low visibility
(4%) Pedestrian & Cyclist Detection Failures: pedestrian, crosswalk
(1%) Static-Obstacle & Sudden Intrusion Collisions: door opening, sudden obstacle
(1%) Infrastructure & Clearance Errors: underpass, vertical clearance
(1%) Other Rare / Long-Tail Cases: wrong-way, extreme/edge-case

Our system successfully discovers a diverse set of interpretable failure clusters from the driving dataset, as listed above and in Fig. 6. Qualitative inspection shows that these clusters correspond to meaningful and recurring traffic incident types, such as `rear-end collisions`, `unsafe lane changes` or `intersection misjudgments`.

Notably, the discovered clusters closely align with the U.S. DOT Volpe Center's pre-crash typology [39], capturing most major failure types observed in real-world driving. This highlights our method's ability to recover semantically grounded failure categories directly from unstructured video data in a way that aligns with expert-defined taxonomies.

*3) **Case Study 3: Vision-Based Indoor Robot Navigation**:* We apply our framework to a vision-based ground robot, navigating unknown indoor office environments [40]. The robot uses a CNN-based policy with a model-based low-level

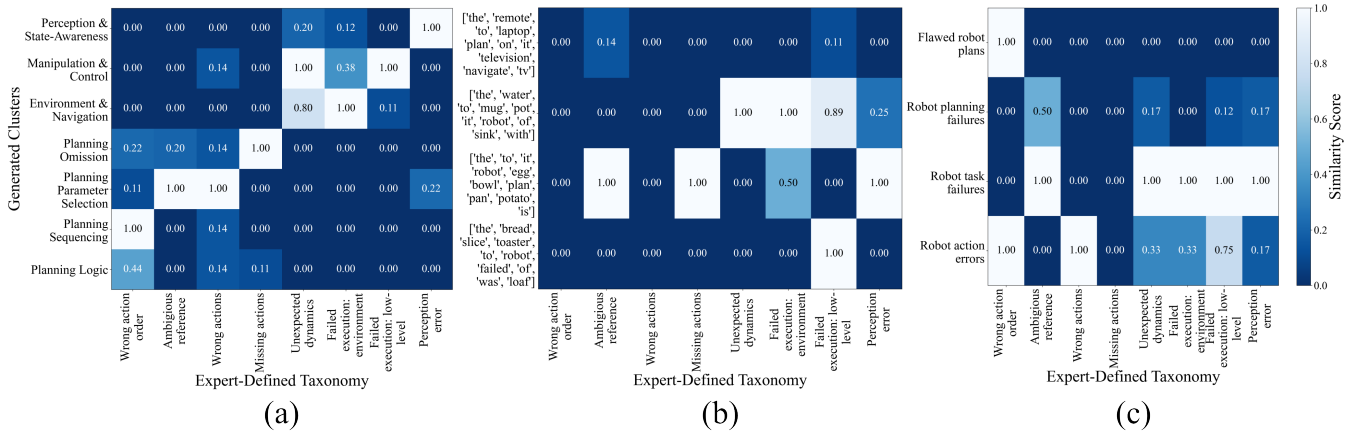

Fig. 4: Heatmaps comparing similarity scores between the RoboFail expert-defined failure taxonomy and the generated clusters by (a) our method, (b) BERTopic, and (c) BERTopic-LLM.

Fig. 6: Real-world car crash failure clusters with examples.

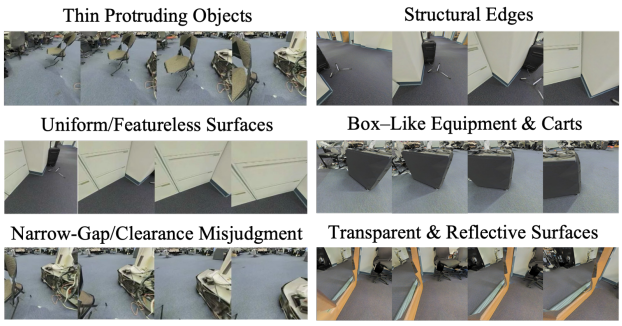

Fig. 7: Major indoor navigation clusters with examples.

controller. It receives RGB images, ego-velocities, and a goal position, and outputs acceleration commands for the robot. We record robot rollouts in the simulated Stanford office environment [41] and extract front-view image sequences. The trajectories resulting in a collision comprise our failure dataset D, which we use for clustering.

Our method discovers a set of interpretable failure clusters from collision trajectories, as listed below and in Fig. 7. Notably, clusters such as `protruding corners`, `white walls`, and `glass doors` were also previously identified by HJ-reachability analysis and manual inspection in [42]. This validates our method's ability to automatically recover known failure types and uncover new semantic patterns.

Vision-Based Indoor Navigation Clusters

(42%) Thin–Protruding Objects: folding chair, thin metal legs
(23%) Uniform/Featureless Surface: featureless wall, white cabinet
(18%) Narrow–Gap/Clearance Misjudgment: narrow passage, insufficient clearance
(10%) Low–Height Clutter & Small Floor Obstacle: backpack, cables
(9%) Box–Like Equipment & Carts: computer tower, server cabinet
(9%) Structural Edges: door frame, wall corner
(4%) Transparent & Reflective Surfaces: glass door, mirror
(6%) Bins & Waste Receptacles: trash bin, recycling bin
(1%) Overhead & Ceiling Fixtures: ceiling fixture, low ceiling

B. *Downstream Safety Enhancement*

We present the results for runtime failure monitoring for the car crash videos and the vision-based indoor navigation systems. We also present targeted data collection and policy fine-tuning results on the latter, as we have access to its operating policy, while we only have a dataset for the former.

*1) **Case Study 1: Real-World Car Crash Videos**:*

a) Failure Monitoring Leveraging Discovered Clusters: We evaluate online failure detection by prompting an MLLM with recent visual history and the discovered failure clusters, and reasoning about any possible near-future collision.

b) Baselines: We compare our approach against *LLM-Based Anomaly Detection (LLM-AD)* methods, such as [44], which provide a few human-written examples of anomalous and nominal objects for a particular system to an LLM, along with the scene description, and ask it to detect any possible anomalies in the current scene. We also compare against *VideoMAE-BC*, the top-performing failure video classifier on the Nexar Crash Prediction Challenge on Kaggle, based on a fine-tuned VideoMAEv2-giant model[43].

c) Results: We compare all methods on a set of 200 held-out in-distribution (IID) driving trajectories. Table II shows that our method consistently achieves the highest F1 score and outperforms baselines in both True Positive Rate (TPR) and False Negative Rate (FNR), showcasing its ability to detect the actual failures both robustly and reliably. These

TABLE II: Failure Detection Metrics (%age) for car crash videos and indoor robot navigation systems. The left and right halves compare metrics for IID and OOD tests, respectively. The last column represents the (average) lead detection time. The best method is highlighted in **bold**.

Method	In-Distribution Trajectories					Out-of-Distribution Trajectories					Time
	TPR	TNR	FPR	FNR	F1	TPR	TNR	FPR	FNR	F1	
Real-World Car Crash Videos											
VideoMAE-BC [43]	52.0	**93.1**	**6.9**	48.0	65.3	18.0	75.0	25.0	82.0	25.2	506.6 ms
LLM-AD	7.1	91.1	8.9	92.9	12.3	35.0	**94.0**	**6.0**	65.0	49.7	166.6 ms
NoContext	42.8	85.3	14.7	57.2	54.1	64.0	80.0	20.0	36.0	69.6	473.3 ms
Ours	**71.4**	72.5	27.5	**28.6**	**71.4**	**83.0**	70.0	30.0	**17.0**	**77.9**	**610 ms**
Vision-Based Indoor Robot Navigation											
ENet-BC [34]	65.0	**100.0**	**0.0**	35.0	**78.8**	**100.0**	6.3	93.7	**0.0**	22.4	1.01 sec
LLM-AD	**83.3**	50.0	50.0	**16.7**	40.0	62.2	60.1	39.9	37.8	27.2	**1.38 sec**
NoContext	51.7	99.6	0.4	48.3	67.4	45.9	**89.0**	**10.1**	54.1	40.5	0.76 sec
Ours	65.0	99.0	1.0	35.0	77.2	67.6	86.2	13.8	32.4	**50.0**	1.21 sec

results also indicate that the system failures are not always the same as anomalies or out-of-distribution (OOD) inputs; IID scenarios, like those specified in our clusters, can also lead to system failures that are hard to capture with LLM-AD. We also test all methods on an unseen dashcam dataset of 200 OOD trajectories. The proposed method demonstrates strong generalization. This suggests our method captures structured semantic patterns beyond dataset-specific cues, unlike VideoMAE-BC, which lacks generalizability.

Finally, we compare the lead failure detection time of different methods in Table II. Our method detects failures earlier than others, indicating a stronger ability to anticipate failures by correlating the scene observation with the clusters.

d) Ablations: We also compare the performance of the proposed failure detection method on removing the failure cluster information from the runtime monitor's prompt (NoContext). All other implementation details, such as processing a history of past observations and CoT reasoning, remain the same as our method. This results in significant performance degradation, highlighting the utility of failure cluster information in better detection across environments.

*2) **Case Study 2: Vision-Based Indoor Robot Navigation**:*

a) Targeted Data Collection and Policy Fine-Tuning: Following Sec. IV-B, we use the discovered clusters to guide expert data collection in targeted regions of the environment. The robot policy is fine-tuned on an augmented dataset containing an additional 40K samples collected in identified failure zones along with the original training data. The failure rate in sampled trajectories drops from 46% to 18%, demonstrating enhanced safety in previously failure-prone situations, whereas fine-tuning with randomly collected additional data only improves the failure rate to 34%. This forms a *closed-loop* pipeline of failure discovery, targeted intervention, and policy refinement for continuously enhancing system safety.

b) Failure Monitoring: We use the same runtime monitoring approach as in Case Study 1. The monitor reasons over the scene, past trajectory, and known failure clusters to preemptively identify any potential failure. As evident in Fig. 7, due to the confined and cluttered nature of the indoor environment, the failure monitor can easily misinterpret static background elements with prior failure contexts, leading to a high false positive rate. To mitigate this, we introduce a simple temporal consistency rule: a failure is flagged only if it persists for three consecutive frames. This helps reduce conservativeness while preserving responsiveness.

c) Baselines: We compare against LLM-AD by providing nominal and anomalous examples relevant to an indoor office environment in the prompt. We additionally compare against *ENet-BC*, a vision-based binary failure classifier based on EfficientNet-B0, trained on labeled collision data from the same environment [34]. Note that, unlike our method and LLM-AD, ENet-BC requires environment-specific training and does not generalize to unseen layouts.

d) Results: On an IID test set of 326 trajectories, our method outperforms all LLM-based baselines in F1 score, as presented in Table II. ENet-BC achieves similar performance to the proposed method, as expected given its environment-specific training. To test generalization, we evaluate on an OOD set of 300 trajectories from a different building. The performance of all methods degrades as expected, but the proposed method maintains the highest F1 score, while ENet-BC fails entirely to generalize. This again demonstrates the generalization capabilities of our method.

Our method also detects failures earlier on average, highlighting its ability to reason about impending collisions before impact. Full metrics are in Table II. Although LLM-AD has a slightly higher average time, it is because it fails entirely and detects everything as anomalous, as evidenced

by its high FPR.

e) Engaging the Safeguard Policy: We integrate our runtime failure monitor with a reactive safeguard controller that activates upon failure detection. Fig. 8 shows an example where the nominal policy leads to collision with a glass door, while the monitor proactively detects the failure and invokes the safeguard controller, enabling successful recovery.

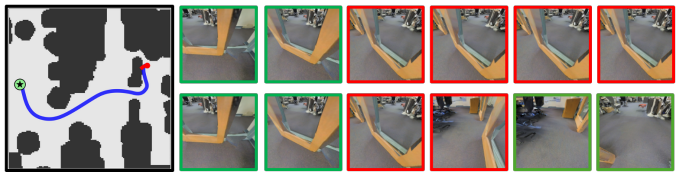

Fig. 8: Robot failing under nominal policy (red) due to misidentifying glass door as traversible, but succeeding under safeguard policy (blue). Red and green borders around first-person view images denote unsafe and safe predictions.

VI. Conclusion

We present a closed-loop framework for interpretable failure analysis in autonomous systems. Our method automatically discovers semantically meaningful failure modes from perception recordings without supervision, organizing them into human-understandable clusters. These clusters enable targeted data collection, policy fine-tuning, and semantic failure detection—supporting scalable and continuous safety improvement. By leveraging the reasoning capabilities of MLLMs, our approach provides a foundation for understanding and responding to failures in unstructured environments.

While our framework enables interpretable and actionable failure analysis, several limitations remain. There is no canonical way to cluster failure trajectories, and different strategies may highlight different insights. Future work could integrate formal methods like STPA or FRAM to complement unsupervised clustering. MLLMs may also produce plausible but incorrect explanations, which could be mitigated through causal or simulation-based validation. Finally, while we evaluate on datasets of $\sim$700 trajectories, our pipeline can readily scale to larger and more temporally extended data.

References

[1] S. Kwon *et al.*, "Image clustering conditioned on text criteria," *ArXiv*, vol. abs/2310.18297, 2023.

[2] M. Liu *et al.*, "Organizing unstructured image collections using natural language," *ArXiv*, vol. abs/2410.05217, 2024.

[3] L. Dunlap *et al.*, "Describing differences in image sets with natural language," *CVPR*, pp. 24 199–24 208, 2023.

[4] Y. Luo *et al.*, "Llm as dataset analyst: Subpopulation structure discovery with large language model," in *ECCV*, 2024.

[5] M. Shen *et al.*, "Sse: Multimodal semantic data selection and enrichment for industrial-scale data assimilation," *ArXiv*, vol. abs/2409.13860, 2024.

[6] S. Terragni *et al.*, "Octis: Comparing and optimizing topic models is simple!" In *EACL SysDemo*, 2021, pp. 263–270.

[7] H. Zhao *et al.*, "Topic modelling meets deep neural networks: A survey," *ArXiv*, 2021.

[8] H. Larochelle *et al.*, "A neural autoregressive topic model," *NeurIPS*, vol. 25, 2012.

[9] Z. Cao *et al.*, "A novel neural topic model and its supervised extension," in *AAAI*, vol. 29, 2015.

[10] J. Qiang *et al.*, "Topic modeling over short texts by incorporating word embeddings," in *PAKDD*, Springer, 2017, pp. 363–374.

[11] M. Shi *et al.*, "We-lda: A word embeddings augmented lda model for web services clustering," in *ICWS*, IEEE, 2017, pp. 9–16.

[12] D. Q. Nguyen *et al.*, "Improving topic models with latent feature word representations," *TACL*, vol. 3, pp. 299–313, 2015.

[13] Y. Liu *et al.*, "Topical word embeddings," in *AAAI*, vol. 29, 2015.

[14] F. Bianchi *et al.*, "Cross-lingual contextualized topic models with zero-shot learning," *ArXiv*, 2020.

[15] A. B. Dieng *et al.*, "Topic modeling in embedding spaces," *TACL*, vol. 8, pp. 439–453, 2020.

[16] L. Thompson *et al.*, "Topic modeling with contextualized word representation clusters," *ArXiv*, 2020.

[17] F. Bianchi *et al.*, "Pre-training is a hot topic: Contextualized document embeddings improve topic coherence," *ArXiv*, 2020.

[18] S. Sia *et al.*, "Tired of topic models? clusters of pretrained word embeddings make for fast and good topics too!" *ArXiv*, 2020.

[19] D. Angelov, "Top2vec: Distributed representations of topics," *ArXiv*, 2020.

[20] M. Grootendorst, "Bertopic: Neural topic modeling with a class-based tf-idf procedure," *ArXiv*, 2022.

[21] F. Indaheng *et al.*, "A scenario-based platform for testing autonomous vehicle behavior prediction models in simulation," *ArXiv*, vol. abs/2110.14870, 2021.

[22] D. J. Fremont *et al.*, "Formal scenario-based testing of autonomous vehicles: From simulation to the real world," *IEEE ITSC*, pp. 1–8, 2020.

[23] T. Dreossi *et al.*, "Systematic testing of convolutional neural networks for autonomous driving," *ArXiv*, vol. abs/1708.03309, 2017.

[24] L. Qiu *et al.*, "Safety-aware imitation learning via mpc-guided disturbance injection," *ArXiv*, vol. abs/2508.03129, 2025.

[25] T. Dreossi *et al.*, "Verifai: A toolkit for the formal design and analysis of artificial intelligence-based systems," in *CAV*, 2019.

[26] T. Zhao *et al.*, "Formal certification methods for automated vehicle safety assessment," *IEEE T-IV*, vol. 8, pp. 232–249, 2022.

[27] S. Ghosh *et al.*, "Counterexample-guided synthesis of perception models and control," *ACC*, pp. 3447–3454, 2019.

[28] Z. Liu *et al.*, "Reflect: Summarizing robot experiences for failure explanation and correction," *ArXiv*, vol. abs/2306.15724, 2023.

[29] J. Duan *et al.*, "Aha: A vision-language-model for detecting and reasoning over failures in robotic manipulation," *ArXiv*, vol. abs/2410.00371, 2024.

[30] W. Lu *et al.*, "Robofac: A comprehensive framework for robotic failure analysis and correction," *ArXiv*, 2025.

[31] J. Wei *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *NeurIPS*, vol. 35, pp. 24 824–24 837, 2022.

[32] Z. Zhao *et al.*, "Calibrate before use: Improving few-shot performance of language models," in *ICML*, PMLR, 2021, pp. 12 697–12 706.

[33] S. Pitis *et al.*, "Boosted prompt ensembles for large language models," *ArXiv*, 2023.

[34] K. Chakraborty *et al.*, "Enhancing safety and robustness of vision-based controllers via reachability analysis," *ArXiv*, 2024.

[35] A. Gupta *et al.*, "Detecting and mitigating system-level anomalies of vision-based controllers," in *ICRA*, IEEE, 2024, pp. 9953–9959.

[36] T. Dreossi *et al.*, "Counterexample-guided data augmentation," *ArXiv*, 2018.

[37] A. Shah *et al.*, "Specification-guided data aggregation for semantically aware imitation learning," *ArXiv*, 2023.

[38] D. Moura *et al.*, "Nexar dashcam collision prediction dataset and challenge," in *CVPR*, 2025, pp. 2583–2591.

[39] W. G. Najm *et al.*, "Pre-crash scenario typology for crash avoidance research," 2007.

[40] S. Bansal *et al.*, "Combining optimal control and learning for visual navigation in novel environments," in *CoRL*, 2020.

[41] I. Armeni *et al.*, "3d semantic parsing of large-scale indoor spaces," *CVPR*, pp. 1534–1543, 2016.

[42] K. Chakraborty *et al.*, "Discovering closed-loop failures of vision-based controllers via reachability analysis," *IEEE RA-L*, vol. 8, no. 5, pp. 2692–2699, 2023.

[43] L. Wang *et al.*, "Videomae v2: Scaling video masked autoencoders with dual masking," in *CVPR*, 2023, pp. 14 549–14 560.

[44] A. Elhafsi *et al.*, "Semantic anomaly detection with large language models," *Autonomous Robots*, vol. 47, pp. 1035–1055, 2023.