LATENTGUARD: Safeguarding Autonomous Driving VLMs via Latent World Logic Repair

Tianyuan Zhang[1], **Maoran Ye**[2], **Yang Qu**[1], **Jiangfan Liu**[1], **Zonglei Jing**[1], **Taichuan Li**[1], **Aishan Liu***[1], **Jiakai Wang**[3], **Yuqing Ma**[1], **Xianglong Liu**[1,3]

[1] SKLCCSE, Beihang University, [2] School of Software, Beihang University, [3] Zhongguancun Laboratory
* Corresponding Author

Abstract

Vision-language models for autonomous driving (AD-VLMs) have been widely adopted to integrate visual perception, language reasoning, and decision-making, yet they still pose serious safety risks under perturbations, misleading instructions, and long-tail traffic interactions. Existing safeguards have shown effectiveness in mitigating unsafe behaviors. However, these safeguards usually introduce external constraints explicitly, making the repair dependent on limited predefined rules or prompt regeneration, which weakens generalization to unseen scenarios and increases generation latency. To address these, we propose LATENTGUARD, a latent world logic repair framework that models safety logic inside the hidden representation space of AD-VLMs. Specifically, LATENTGUARD learns logic-induced feature shifts from unsafe outputs and their counterfactual safe alternatives, enabling the model to repair unsafe hidden states toward safer world-consistent representations. To further improve generalization, we introduce conditional latent logic modeling and future-consistency regularization, which encourage repaired states to follow safer latent world evolution beyond the current output. Experiments on VRU-Accident and NAVSIM show that LATENTGUARD reduces unsafe response rate from 32.6% to 18.8% and improves PDMS from 82.3 to 84.6. Closed-loop and human-in-the-loop case studies further validate that LATENTGUARD can repair unsafe AD-VLMs responses in realistic driving interactions while preserving driving behavior.

1 Introduction

Vision-language models (VLMs) have been widely adopted in autonomous driving (AD), forming AD-VLMs as a promising direction for visual perception, language-conditioned reasoning, and planning-oriented decision making [Sima *et al.*, 2024; Ma *et al.*, 2024; Gopalkrishnan *et al.*, 2024]. Given driving scenes and textual instructions, AD-VLMs are expected to recognize traffic participants, infer scene-level risks, and generate driving-relevant responses

Figure 1: Illustration of LATENTGUARD. LATENTGUARD repairs unsafe AD-VLMs responses by shifting the latent representation from an unsafe state to a safer world-consistent state.

or actions. However, their deployment still faces serious safety challenges. Under adversarial perturbations, misleading instructions, rare vulnerable-road-user interactions, and long-tail traffic scenarios, AD-VLMs may produce outputs that appear plausible but are unsafe in the dynamic driving world [Zhang *et al.*, 2025a; Zhang *et al.*, 2026]. These failures are often not merely output-level mistakes; they can originate from latent semantic, causal, or interaction-level misunderstandings before decoding.

To address these problems, safeguards have become an important solution for mitigating unsafe behaviors [Ravichandran *et al.*, 2025; Wang *et al.*, 2025], especially logic-based safeguards that introduce interpretable and controllable safety constraints [Zhang *et al.*, 2025a; Zhang *et al.*, 2026]. Existing methods usually protect models by injecting explicit rules, checking generated actions, or regenerating prompts and outputs according to external safety logic. Although effective in known scenarios, they are limited by predefined rule coverage and rigid correction patterns. Real-world AD safety involves implicit intentions, multi-agent negotiation, and context-dependent causal relations, which are difficult to exhaustively enumerate. Moreover, prompt or output regeneration introduces extra inference overhead and may still fail to repair the hidden representation that causes unsafe responses.

This motivates a latent-space view of safety repair for AD-VLMs. Instead of treating safety as an external rule set after generation, we argue that safety logic should be modeled

inside the hidden representation space. Inspired by world-model studies showing that latent representations can encode scene dynamics, agent interactions, and traffic-level regularities [Hu *et al.*, 2022; Hu *et al.*, 2023; Wang *et al.*, 2024; Zheng *et al.*, 2024], we define these safety-relevant hidden-state regularities as *latent world logic*. From this view, repairing an unsafe AD-VLMs output requires learning how the feature representation changes when safety logic is introduced into the reasoning process.

In this paper, we propose LATENTGUARD, a latent world logic repair framework for safeguarding AD-VLMs. Instead of injecting explicit rules during inference, LATENTGUARD introduces safety logic implicitly by learning the feature-layer changes induced by counterfactual safe correction. During training, unsafe outputs are paired with safe alternatives obtained from safety correction, expert annotation, simulator verification, or safe future supervision, and the corresponding hidden-state differences are used as feature-level repair supervision. This converts explicit or counterfactual safety evidence into latent repair knowledge, allowing LATENTGUARD to model how unsafe representations should move toward safer and world-consistent states. Based on this supervision, a conditional latent logic module learns diverse repair patterns and regenerates executable hidden-state repair signals from the scene, instruction, original output, and failure cue. At inference time, LATENTGUARD only uses the learned prior and repair decoder, without requiring explicit rule search, counterfactual safe alternatives, or replacement of the original decoder. To improve generalization beyond current-output repair, we further introduce a training-only future-consistency regularizer that encourages repaired latent states to align with safer world evolution. We evaluate LATENTGUARD on VRU-Accident and NAVSIM, where LATENTGUARD reduces unsafe response rate from 32.6% to 18.8% and improves PDMS from 82.3 to 84.6. We also conduct physical-world case studies to examine the effectiveness of latent repair. Our **contributions** are summarized as:

- We formulate AD-VLMs safety repair as a *latent world logic repair* problem, moving beyond explicit rule enumeration, prompt regeneration, and output correction.
- We propose LATENTGUARD, a latent world logic repair framework that learns logic-induced feature shifts, regenerates hidden-state repair signals, and regularizes them with future consistency.
- Comprehensive experiments on accident-centric scenarios, common driving scenarios, and physical-world case studies demonstrate the effectiveness and generalization.

2 Related Work

Safeguards for AD-VLMs. Safety guards aim to prevent foundation models [Brohan *et al.*, 2023; Kim *et al.*, 2024] from producing unsafe behaviors through rule filters, external checkers, action-level correction, or model-level guardrails [Zhou *et al.*, 2025]. RoboGuard converts safety rules into executable temporal-logic constraints [Ravichandran *et al.*, 2025; Gokhale *et al.*, 2025], while RoboSafe uses executable safety logic with reflective and predictive reasoning to mitigate contextual risks [Wang *et al.*, 2025]. GuardAgent employs a separate guard agent for knowledge-enabled safety inspection [Xiang *et al.*, 2024], and SafeMind studies cascaded safety modules across task understanding, perception, planning, and action generation [Chen *et al.*, 2025]. HomeGuard and EMBGuard extend embodied safeguards to contextual risk detection and hazard-aware planning [Lu *et al.*, 2026; Choi *et al.*, 2026]. In autonomous driving, recent work further introduces logic-based safeguards for physical-world agents. SafeAuto enhances VLM-based driving with structured safety knowledge and uses first-order logic with probabilistic graphical reasoning to verify predicted actions [Zhang *et al.*, 2025a]. GuardAD formulates driving safety as evolving Markovian safety logic and revises unsafe actions for autonomous-driving models [Zhang *et al.*, 2026]. These methods show the effectiveness of explicit logic for safeguarding physical-world AI systems. However, they mainly operate at the rule, symbolic-state, output, or action-revision level, and rely on manually specified logic, external checking, or predefined safety predicates. As a result, existing safeguard methods still leave open how safety regularities can supervise feature-level repair of unsafe AD-VLMs hidden states [Meng *et al.*, 2022; Zou *et al.*, 2023; Turner *et al.*, 2023; Wu *et al.*, 2024].

World Models for Autonomous Driving. World-model research increasingly emphasizes learning compact world-state abstractions in latent space rather than reconstructing all low-level observations. LeCun *et al.*'s Joint-Embedding Predictive Architecture (JEPA) advocates world modeling by predicting missing or future representations in an abstract embedding space [LeCun, 2022], and later works, such as I-JEPA and V-JEPA, further show that latent feature prediction can capture semantic and temporal regularities from images and videos [Assran *et al.*, 2023; Bardes *et al.*, 2024]. This latent-space perspective has also been explored in AD world models [Shi *et al.*, 2025; Zheng *et al.*, 2025; Jia *et al.*, 2026; Wang *et al.*, 2026]. MILE learns latent dynamics and driving policies from video observations [Hu *et al.*, 2022], GAIA-1 and DriveDreamer generate realistic or controllable future driving scenes [Hu *et al.*, 2023; Wang *et al.*, 2024], DriveDreamer-2 introduces language-conditioned driving video generation [Zhao *et al.*, 2025], and OccWorld models 3D occupancy evolution for planning-oriented prediction [Zheng *et al.*, 2024]. These works suggest that latent driving representations can encode agent motion, scene dynamics, road topology, interactions, and temporal consistency. However, existing world-model studies mainly focus on prediction, generation, simulation fidelity, or planning utility, while their latent-space insights have rarely been used to derive logic-induced representation shifts for repairing unsafe behaviors produced by AD-VLMs [Tian *et al.*, 2024; Mao *et al.*, 2023] under adversarial perturbations, misleading instructions, or high-risk traffic interactions.

3 Method

3.1 Overview

Given a visual driving input $\mathbf{x}$ and a language instruction $\mathbf{q}$, a base AD-VLMs produces an output $\mathbf{y}$ from its hidden rep-

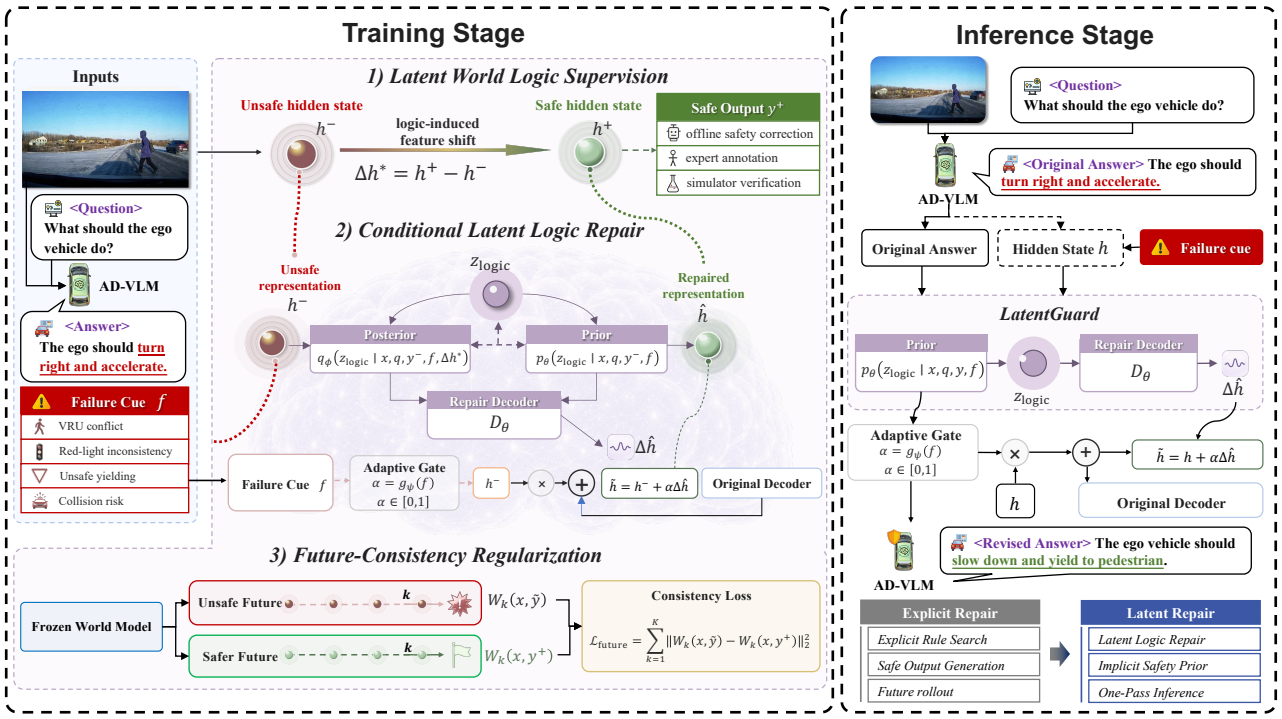


Figure 2: Overall framework. LATENTGUARD learns logic-induced feature shifts from unsafe responses and counterfactual safe alternatives, regenerates hidden-state repair signals through a conditional latent logic module, and regularizes the repaired state with safer future evolution.

resentation $\mathbf{h}$. Instead of filtering or rewriting the output after generation, LATENTGUARD repairs the hidden representation that leads to unsafe behavior. The key idea is to learn *latent world logic* from counterfactual correction: when an unsafe response is changed into a safer alternative, the corresponding feature-level change indicates how safety logic should act inside the model.

Given the original output, LATENTGUARD extracts a failure cue $\mathbf{f}$ to summarize the unsafe behavior, such as vulnerable-road-user conflict, unsafe yielding, misleading-instruction compliance, collision risk, or red-light inconsistency. The overall repair process is

$$(\mathbf{x}, \mathbf{q}, \mathbf{y}, \mathbf{f}) \rightarrow \Delta\hat{\mathbf{h}} \rightarrow \tilde{\mathbf{h}} \rightarrow \tilde{\mathbf{y}}, \tag{1}$$

where $\Delta\hat{\mathbf{h}}$ is the predicted hidden-state repair signal, $\tilde{\mathbf{h}}$ is the repaired representation, and $\tilde{\mathbf{y}}$ is decoded by the original AD-VLMs head. Thus, LATENTGUARD performs safety repair inside the latent reasoning state without explicit rule enumeration, output regeneration, or decoder replacement.

3.2 Latent World Logic Supervision

The first step is to learn what a safe latent repair should look like. When the original output is unsafe, we denote it as $\mathbf{y}^-$ and denote its hidden representation as $\mathbf{h}^-$. For each unsafe response, we construct or retrieve a counterfactual safe alternative $\mathbf{y}^+$ that keeps the original task intent while removing the unsafe behavior. This safe alternative can be obtained from explicit safety logic, expert correction, simulator-verified decisions, safe labels, or safe future supervision.

The safe alternative is not used as a fixed output template. Instead, it is used to reveal how the internal representation should change when safety logic is introduced. Let $\mathbf{h}^+$ be the hidden representation associated with $\mathbf{y}^+$, obtained by teacher-forcing the safe response, projecting the safe alternative into the selected hidden layer, or using an aligned lightweight projection module. The logic-induced feature shift is defined as

$$\Delta\mathbf{h}^* = \mathbf{h}^+ - \mathbf{h}^-. \tag{2}$$

This target describes the feature-level effect of safety correction. In other words, explicit or counterfactual safety evidence is distilled into a latent repair target. After training on many such pairs, the repair module learns how unsafe hidden states should move toward safer and world-consistent states, without requiring counterfactual alternatives.

To avoid learning trivial conservative behavior, we only keep counterfactual pairs that satisfy two conditions. First, the safe alternative must reduce driving risk compared with the unsafe response. Second, it must preserve the original task semantics, rather than simply refusing, stopping in all cases, or producing irrelevant safe-sounding text. This filtering is important for AD-VLMs because overly conservative outputs may appear safe locally but can destroy planning utility or violate the instruction. The retained pairs therefore provide supervision for meaningful safety repair rather than generic output suppression.

3.3 Conditional Latent Logic Repair

Different unsafe cases may require different repair patterns. A vulnerable-road-user conflict may require yielding, slowing down, shifting attention to the pedestrian, or rejecting a misleading instruction. A red-light inconsistency may require

revising scene understanding, correcting action preference, or suppressing an unsafe instruction-following tendency. Therefore, LATENTGUARD does not learn a single deterministic repair vector for all failures. Instead, it uses a conditional latent logic module to model diverse repair patterns.

The repair module takes the scene, instruction, original output, and failure cue as input, and predicts a hidden-state repair signal $\Delta\hat{\mathbf{h}}$. Internally, the module uses a compact latent repair code to capture the type of safety correction needed for the current case. During training, this latent code is guided by the teacher shift $\Delta\mathbf{h}^*$; during inference, it is inferred only from the current context. This design allows the model to learn diverse latent repair modes while avoiding rule search.

The predicted repair signal is supervised by the logic-induced feature shift:

$$\mathcal{L}_{logic} = \left\| \Delta\mathbf{h}^* - \Delta\hat{\mathbf{h}} \right\|_2^2 + \beta\mathcal{L}_{KL}, \tag{3}$$

where the first term learns the feature-level repair direction, and $\mathcal{L}_{KL}$ regularizes the training-time latent repair code toward the inference-time latent distribution. This regularization makes the repair module usable when the safe alternative and teacher shift are unavailable. The coefficient β controls the strength of this latent distribution alignment.

After obtaining the repair signal, LATENTGUARD updates the hidden representation as

$$\tilde{\mathbf{h}} = \mathbf{h}^- + \alpha\Delta\hat{\mathbf{h}}, \tag{4}$$

where α is a fixed or learned repair strength. A small gate can predict α from the failure cue, so that safe or low-risk samples receive little intervention while high-risk samples receive stronger repair. The repaired output is then generated by the original AD-VLMs decoder:

$$\tilde{\mathbf{y}} = D_\Theta(\tilde{\mathbf{h}}). \tag{5}$$

This keeps the original decoder unchanged and makes LATENTGUARD a lightweight safety adapter.

3.4 Future-Consistent Regularization

A repaired output should not only look safer at the current step; it should also be compatible with safer future driving evolution. For example, a response may mention slowing down, but its latent state may still preserve an unsafe interaction pattern that leads to collision-prone future behavior. To reduce such superficial repairs, LATENTGUARD introduces a training-only future-consistency regularizer.

We use a frozen rollout source $\mathcal{W}$. This rollout source is not trained or deployed as part of LATENTGUARD . It only provides future-oriented supervision during training. Given the repaired output $\tilde{\mathbf{y}}$ and the counterfactual safe alternative $\mathbf{y}^+$, $\mathcal{W}$ produces two future latent trajectories: one induced by the repaired response and one induced by the safe alternative. The future-consistency loss is

$$\mathcal{L}_{future} = \sum_{k=1}^{K} \left\| \mathcal{W}_k(\mathbf{x}, \tilde{\mathbf{y}}) - \mathcal{W}_k(\mathbf{x}, \mathbf{y}^+) \right\|_2^2, \tag{6}$$

where K is the rollout horizon and $\mathcal{W}_k(\cdot)$ denotes the future latent state at step k. This loss encourages the repaired representation to move toward the future region associated with safe driving behavior. It gives temporal meaning to the latent repair, rather than treating safety as a single-step correction.

In addition to future consistency, we use two lightweight regularization terms. The risk regularizer encourages the repaired output to have lower risk than the unsafe response. The conservative consistency regularizer discourages unnecessary hidden-state changes when the original response is already safe. Together, these terms prevent two common failure modes: under-repair, where the output becomes superficially safe but remains dynamically unsafe, and over-repair.

3.5 Training and Inference

The full objective is

$$\mathcal{L} = \mathcal{L}_{task} + \lambda_l\mathcal{L}_{logic} + \lambda_f\mathcal{L}_{future} + \lambda_r\mathcal{L}_{risk} + \lambda_c\mathcal{L}_{cons}. \tag{7}$$

Here, $\mathcal{L}_{task}$ supervises the final answer, action, or planning-oriented response, while the other terms correspond to logic-induced feature-shift learning, future-consistency regularization, risk reduction, and conservative consistency.

During training, we first freeze or lightly adapt the base AD-VLMs , then construct unsafe-safe pairs and extract their hidden-state differences as repair targets. The conditional latent logic module is trained with Eq. (7) to predict hidden-state repair signals, while the original decoder can remain frozen. At inference time, LATENTGUARD only uses the scene $\mathbf{x}$, instruction $\mathbf{q}$, original output $\mathbf{y}$, and failure cue $\mathbf{f}$. It predicts a repair signal, updates the hidden representation, and decodes the repaired output through the original AD-VLMs head, without using safe alternatives, teacher shifts, explicit rule search, or future rollout.

4 Experiments

4.1 Experiment Setting

Datasets. We evaluate LATENTGUARD on two datasets. VRU-Accident [Kim *et al.*, 2025] is used for accident-centric evaluation, focusing on high-risk traffic cases. NAVSIM [Dauner *et al.*, 2024] is used for common-scene evaluation, focusing on planning-oriented driving performance under ordinary traffic scenarios.

Models. We evaluate LATENTGUARD on four representative AD-VLMs: DriveLM [Sima *et al.*, 2024], Dolphins [Ma *et al.*, 2024], EM-VLM4AD [Gopalkrishnan *et al.*, 2024], and DriveWorld-VLA [Jia *et al.*, 2026]. DriveWorld-VLA is included as a stronger VLA-style model with latent world modeling ability. For each model, LATENTGUARD is applied to the selected multimodal reasoning or planning layer, while the base model and original output head are kept frozen.

Baselines. We compare LATENTGUARD with the vanilla model and representative safeguard methods, including RoboFactory (RF) [Qin *et al.*, 2025], Code-as-Monitor (CM) [Zhou *et al.*, 2025], SafeAuto (SA) [Zhang *et al.*, 2025a], and GuardAD (GA) [Zhang *et al.*, 2026]. We also include deterministic latent repair (Det. Repair) in ablation studies, which removes conditional latent modeling and predicts a single hidden-state repair direction.

Implementation Details. For each scene $\mathbf{x}$ and instruction $\mathbf{q}$, the base model produces an output $\mathbf{y}$ and hidden representation $\mathbf{h}$, and a failure cue $\mathbf{f}$ is constructed to indicate the unsafe factor. During training, counterfactual safe alternatives

Table 1: Main results on VRU-Accident. "+Method" denotes adding the corresponding safeguard to the Vanilla model. URR denotes unsafe response rate, RRR denotes risk recognition rate, RSR denotes repair success rate, and FC denotes future consistency.

Model	Method	Acc. ↑	URR ↓	RRR ↑	RSR ↑	FC ↑
DriveLM	Vanilla	61.8	36.4	56.7	–	52.1
DriveLM	+RF	60.9	33.8	59.5	39.6	53.0
DriveLM	+CM	60.6	32.9	60.8	41.2	53.8
DriveLM	+SA	62.4	29.7	64.9	48.5	57.9
DriveLM	+GA	63.1	24.8	68.6	58.3	62.4
DriveLM	**+LATENTGUARD**	**64.0**	**20.6**	**71.4**	**63.7**	**70.9**
Dolphins	Vanilla	64.5	33.7	59.4	–	55.2
Dolphins	+RF	63.6	31.6	61.3	42.8	56.4
Dolphins	+CM	63.2	30.8	62.6	44.0	56.1
Dolphins	+SA	65.0	27.9	66.8	51.7	60.5
Dolphins	+GA	65.4	23.0	70.5	61.4	65.8
Dolphins	**+LATENTGUARD**	**66.3**	**19.2**	**73.1**	**66.5**	**74.0**
EM-VLM4AD	Vanilla	66.7	31.2	62.5	–	57.0
EM-VLM4AD	+RF	65.9	29.4	64.1	44.5	57.8
EM-VLM4AD	+CM	65.6	28.7	65.3	45.9	58.6
EM-VLM4AD	+SA	67.1	25.8	69.2	53.9	62.7
EM-VLM4AD	+GA	67.6	21.1	72.8	64.2	67.6
EM-VLM4AD	**+LATENTGUARD**	**68.5**	**18.3**	**75.0**	**69.1**	**75.2**
DriveWorld-VLA	Vanilla	70.4	28.9	65.8	–	60.1
DriveWorld-VLA	+RF	69.6	27.0	67.2	47.6	60.9
DriveWorld-VLA	+CM	69.1	26.4	68.1	48.4	61.8
DriveWorld-VLA	+SA	70.8	23.8	71.9	56.8	66.0
DriveWorld-VLA	+GA	71.3	19.4	75.6	67.5	70.5
DriveWorld-VLA	**+LATENTGUARD**	**72.1**	**16.9**	**78.0**	**72.3**	**78.2**

are obtained through an offline pre-search guided by multiple safeguard methods, such as SafeAuto and GuardAD. We select candidates that reduce unsafe behavior while preserving the original task intent, and use their hidden-state differences from unsafe responses as logic-induced feature-shift supervision. During inference, this pre-search is removed.

Metrics. On VRU-Accident, we report task accuracy (Acc.), Unsafe Response Rate (URR), Risk Recognition Rate (RRR), Repair Success Rate (RSR), and Future Consistency (FC). RSR is reported only for methods with safeguard or repair intervention, and is therefore not applicable to Vanilla models. On NAVSIM, we report PDMS, No Collision (NC), Drivable Area Compliance (DAC), time-to-collision safety score (TTC), Comfort (Comf.), and Ego Progress (EP).

4.2 Main Results

Results on Accident Scenarios. Tab. 1 reports the comparison on VRU-Accident dataset.

❶ Across the four models, LATENTGUARD reduces the average URR from 32.6% to 18.8%, achieving a 13.8-point absolute reduction and a 42.4% relative reduction over Vanilla. Meanwhile, the average Acc. increases from 65.9 to 67.7. This indicates that LATENTGUARD does not simply suppress responses or over-conservatively reject risky cases, but improves safety while preserving task correctness.

❷ Compared with RF and CM, whose average URR values are 30.5% and 29.7%, LATENTGUARD achieves a much lower average URR of 18.8%. It also improves the average RRR to 74.4, compared with 63.0 for RF and 64.2 for CM. This suggests that detecting unsafe outputs after generation is less effective than repairing the hidden reasoning state.

❸ Compared with SA and GA, LATENTGUARD further improves average RSR from 52.7 and 62.9 to 67.9, and improves average FC from 61.8 and 66.6 to 74.6. These gains suggest that logic-induced feature shifts provide more effective and temporally consistent repair than directly applying structured safety knowledge or revising the final action.

❹ The improvement is consistent across DriveLM, Dolphins, EM-VLM4AD, and DriveWorld-VLA. On the strongest DriveWorld-VLA model, LATENTGUARD still reduces URR from 28.9% to 16.9% and improves FC from 60.1 to 78.2. This shows that latent world logic repair remains useful even when the base model already has stronger world-level representation ability.

Table 2: Main results on NAVSIM. "+Method" denotes adding the corresponding safeguard or repair module to the Vanilla model. PDMS is the main planning score. NC, DAC, TTC, Comf., and EP evaluate collision avoidance, drivable-area compliance, time-to-collision safety, ride comfort, and ego progress.

Model	Method	PDMS ↑	NC ↑	DAC ↑	TTC ↑	Comf. ↑	EP ↑
DriveLM	Vanilla	76.8	92.0	90.8	84.5	99.4	69.8
DriveLM	+RF	76.1	93.3	91.4	86.0	99.0	68.5
DriveLM	+CM	75.8	93.8	91.7	86.6	98.8	68.0
DriveLM	+SA	77.6	94.6	92.6	88.2	99.1	69.1
DriveLM	+GA	78.7	95.4	93.3	89.5	99.2	70.0
DriveLM	**+LATENTGUARD**	**79.6**	**96.2**	**93.8**	**90.4**	**99.5**	**70.9**
Dolphins	Vanilla	79.0	94.0	92.1	86.0	99.5	72.5
Dolphins	+RF	78.4	95.0	92.8	87.4	99.1	71.0
Dolphins	+CM	78.1	95.3	93.0	88.0	98.9	70.5
Dolphins	+SA	79.9	96.0	93.8	89.4	99.3	72.0
Dolphins	+GA	80.9	96.5	94.4	90.5	99.4	72.8
Dolphins	**+LATENTGUARD**	**81.8**	**97.0**	**95.0**	**91.6**	**99.6**	**73.6**
EM-VLM4AD	Vanilla	82.0	95.8	93.7	88.8	99.6	75.6
EM-VLM4AD	+RF	81.4	96.4	94.1	90.0	99.2	74.3
EM-VLM4AD	+CM	81.1	96.8	94.5	90.4	99.0	73.7
EM-VLM4AD	+SA	82.8	97.1	95.0	91.5	99.4	75.2
EM-VLM4AD	+GA	83.7	97.5	95.6	92.4	99.5	76.1
EM-VLM4AD	**+LATENTGUARD**	**84.5**	**97.8**	**96.0**	**93.1**	**99.7**	**77.0**
DriveWorld-VLA	Vanilla	91.3	99.1	98.2	96.1	100.0	85.9
DriveWorld-VLA	+RF	90.8	99.2	98.3	96.5	99.8	84.6
DriveWorld-VLA	+CM	90.6	99.3	98.4	96.8	99.7	84.1
DriveWorld-VLA	+SA	91.6	99.3	98.5	96.9	99.9	85.5
DriveWorld-VLA	+GA	92.0	99.4	98.6	97.1	100.0	86.1
DriveWorld-VLA	**+LATENTGUARD**	**92.4**	**99.5**	**98.8**	**97.4**	**100.0**	**86.7**

Results on Common Scenarios. Tab. 2 reports the comparison on NAVSIM.

❶ Across the four models, LATENTGUARD improves the average PDMS from 82.3 to 84.6, with consistent gains on every evaluated model. This indicates that latent repair improves safety-oriented driving behavior while preserving common-scene planning utility.

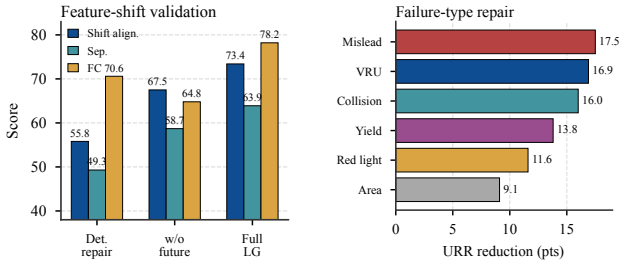
❷ The improvements are more evident on safety-related metrics. Compared with Vanilla, LATENTGUARD increases the average NC from 95.2 to 97.6 and the average TTC from 88.9 to 93.1. These results show that LATENTGUARD improves collision avoidance and time-to-collision safety, which is consistent with its goal of repairing unsafe latent states.

❸ External monitoring methods improve some safety scores but tend to reduce driving progress. For example, +RF and +CM reduce the average EP from 76.0 to 74.6 and 74.1, respectively. In contrast, LATENTGUARD improves the average EP to 77.1, suggesting that hidden-state repair avoids the rigid intervention pattern of output-level safeguards.

❹ On DriveWorld-VLA, LATENTGUARD improves PDMS from 91.3 to 92.4, NC from 99.1 to 99.5, TTC from 96.1 to

Table 3: Component ablation on DriveWorld-VLA. URR, RSR, and FC are reported on VRU-Accident, while PDMS, NC, and EP are reported on NAVSIM.

Variant	URR ↓	RSR ↑	FC ↑	PDMS ↑	NC ↑	EP ↑
Full LATENTGUARD	16.9	72.3	78.2	92.4	99.5	86.7
w/o logic-induced feature shift	23.7	55.1	68.4	91.0	99.0	85.6
w/o latent distribution	19.8	64.0	72.5	91.7	99.2	86.1
w/o failure cue	22.4	58.7	70.1	91.3	99.1	85.8
w/o future consistency	18.9	68.2	64.8	92.0	99.3	86.4
w/o repair gate	17.5	70.4	74.6	90.6	99.4	84.9



(a) Feature-shift validation. (b) Failure-type repair.

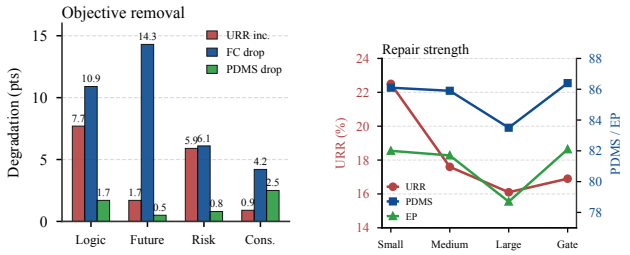
Figure 3: Mechanism and failure-type visualization. Full LATENTGUARD obtains the strongest feature-shift alignment and future consistency among the repair variants, and its URR reduction covers multiple accident-related failure types.

97.4, and EP from 85.9 to 86.7. This indicates that LATENTGUARD can complement latent-world-model-based models.

4.3 Ablation Studies

Component Ablation. We ablate the main components of LATENTGUARD on DriveWorld-VLA and report results on both VRU-Accident and NAVSIM. As shown in Tab. 3, removing logic-induced feature-shift supervision causes the largest safety degradation, increasing URR from 16.9% to 23.7% and reducing RSR from 72.3 to 55.1. Removing the latent distribution or failure cue also weakens repair, showing that diverse correction modes and explicit risk context are both important. Future consistency mainly affects temporal reliability, with FC dropping from 78.2 to 64.8. Removing the repair gate only slightly affects URR, but reduces PDMS and EP, indicating that adaptive repair strength is necessary for preserving common-scene utility.

Feature-Shift and Failure-Type Analysis. Fig. 3 examines whether the predicted repair signal matches the intended logic-induced feature shift and whether the repair effect generalizes across accident types. Full LATENTGUARD improves shift alignment from 55.8 to 73.4 over deterministic repair and increases safe-unsafe separation from 49.3 to 63.9. Removing future consistency keeps reasonable alignment but reduces FC from 78.2 to 64.8, indicating that feature-shift matching alone is insufficient for temporally reliable repair. Across failure categories, LATENTGUARD reduces URR most strongly on misleading instruction, VRU conflict, and collision-prone decisions, while the smaller gain on drivable-area violations suggests that geometry-sensitive errors still require stronger spatial grounding.



(a) Loss-term removal. (b) Repair-strength trade-off.

Figure 4: Ablation diagnostics on DriveWorld-VLA. (a) Removing each loss term is reported as degradation relative to the full objective. (b) Repair strength controls the safety-utility trade-off; the learned gate preserves utility while maintaining low URR.

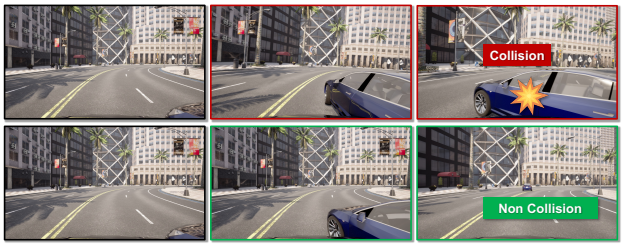


Figure 5: Closed-loop case in CARLA. The top row shows Vanilla DriveLM producing an unsafe maneuver that leads to a collision, while the bottom row shows DriveLM+LATENTGUARD repairing the unsafe latent state and generating a collision-avoiding decision.

Objective and Sensitivity Analysis. As shown in Fig. 4, removing $\mathcal{L}_{logic}$ causes the largest URR increase, confirming the importance of feature-shift reconstruction. Removing $\mathcal{L}_{future}$ produces the largest FC drop, indicating its role in temporal consistency. Removing $\mathcal{L}_{cons}$ mainly hurts PDMS, which shows that conservative consistency helps prevent over-repair on safe samples. For repair strength, a small α under-corrects unsafe states, whereas a large α lowers URR but hurts PDMS and EP.

5 Case Study

5.1 Closed-loop Simulation Evaluation

We evaluate LATENTGUARD on Bench2ADVLM [Zhang *et al.*, 2025b] in a closed-loop CARLA setting, where model predictions affect future observations and long-horizon driving outcomes. We use CARLA 0.9.15 with a LLaMA-based DriveLM model, and compare Vanilla DriveLM with DriveLM+LATENTGUARD under the same perception inputs, decoding settings, and simulation configurations.

LATENTGUARD improves the route success rate from 8.46% to 13.84%, corresponding to a 63.6% relative gain, and reduces the collision rate from 65.0% to 34.0%. As shown in Fig. 5, Vanilla DriveLM produces an unsafe maneuver in a dynamic interaction scene and eventually collides with the surrounding vehicle. With LATENTGUARD enabled, the unsafe hidden state is repaired toward a safer interaction-aware representation, leading to a collision-avoiding decision. This result suggests that latent world logic repair can improve

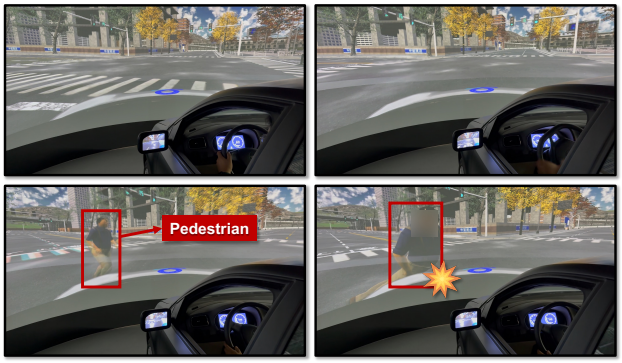


Figure 6: Human-in-the-loop evaluation. The frames show a representative pedestrian-rush scenario. Under the same observations, Vanilla DriveLM outputs *Drive*, while DriveLM+LATENTGUARD revises the unsafe response to *Stop*.

closed-loop driving behavior under compounding scene dynamics, rather than only correcting isolated outputs.

5.2 Human-in-the-loop Evaluation

We further conduct a human-in-the-loop evaluation with a vehicle-simulator platform. A human driver operates the vehicle in a risky pedestrian-crossing scenario, where a pedestrian suddenly rushes into the ego lane. We record synchronized observations and collision outcomes, and then feed the same observations to Vanilla DriveLM and DriveLM+LATENTGUARD for action prediction. Following prior evaluation practice [Sato *et al.*, 2021], a trial is counted as a collision if the model fails to output a safe stop within a 2.5s window before the annotated collision time.

Over 20 trials per condition, the collision rates are 20%, 55%, and 25% for human driving, Vanilla DriveLM, and DriveLM+LATENTGUARD, respectively. Compared with Vanilla DriveLM, LATENTGUARD reduces the collision rate by 30 percentage points. As illustrated in Fig. 6, the pedestrian-rush scenario is highly collision-prone; under the same scene observations, Vanilla DriveLM outputs *Drive*, whereas DriveLM+LATENTGUARD repairs the unsafe latent state and outputs *Stop*. These results show that LATENTGUARD can effectively suppress unsafe driving decisions in human-in-the-loop interactions and narrow the gap between AD-VLMs decisions and human driving behavior.

6 Discussion

Relation to explicit safety logic. LATENTGUARD does not discard explicit safety logic; instead, it changes how such logic is used. During training, explicit rules, expert correction, simulator checks, and safe future supervision provide counterfactual safe alternatives for learning latent repair targets. During inference, however, LATENTGUARD no longer enumerates rules or searches for safe alternatives. This makes latent repair less dependent on rigid rule templates, while still allowing explicit verification to remain as a fallback.

Closed-loop effect. The case studies suggest that latent repair can bring larger benefits in interactive settings than in single-step evaluation. In offline benchmarks, LATENTGUARD reduces average unsafe response rate from 32.6% to 18.8%. In closed-loop CARLA evaluation, it improves route success from 8.46% to 13.84% and reduces collision rate from 65.0% to 34.0%. In the human-in-the-loop pedestrian-rush scenario, DriveLM+LATENTGUARD reduces the collision rate from 55% to 25%. These results indicate that correcting an unsafe latent state can prevent error accumulation over future observations and actions.

Table 4: Efficiency analysis on VRU-Accident. We report inference time (seconds) and standard deviation σ over 10 runs. The fastest safeguard method in each row is in bold.

Model	Vanilla	+RF		+CM		+SA		+GA		+LATENTGUARD	
	Time	Time	σ	Time	σ	Time	σ	Time	σ	Time	σ
DriveLM	**1.36**	1.89	0.16	1.76	0.25	1.79	0.24	1.77	0.25	1.38	0.12
Dolphins	**2.75**	4.23	0.48	3.88	0.67	3.49	0.59	3.54	0.56	2.76	0.31

Safety-utility balance. A key requirement for AD-VLMs safeguards is to reduce risk without making the model overly conservative. LATENTGUARD lowers unsafe response rate while preserving normal driving utility: average PDMS improves from 82.3 to 84.6, and ego progress increases from 76.0 to 77.1. The repair gate is important for this balance. Removing it only slightly changes URR from 16.9% to 17.5%, but reduces PDMS from 92.4 to 90.6 and ego progress from 86.7 to 84.9. This suggests that adaptive repair strength helps LATENTGUARD intervene on risky states while avoiding unnecessary changes on safe cases.

Efficiency Analysis We further evaluate the inference efficiency of different safeguards on DriveLM and Dolphins. Following prior safeguard evaluation, we report the average inference time and standard deviation over 10 runs on VRU-Accident. As shown in Tab. 4, explicit safeguards introduce additional inference cost due to rule checking, prompt regeneration, or action-level revision. In contrast, LATENTGUARD performs repair through a lightweight latent adapter and does not require online rule enumeration, counterfactual search, or future rollout during inference. Therefore, its additional computation is negligible after feature extraction and decoder reuse. The rounded inference time of LATENTGUARD is almost same as the Vanilla, *i.e.*, 1.38s on DriveLM and 2.76s on Dolphins. These results suggest that LATENTGUARD can improve safety without increasing inference latency.

7 Conclusion

This paper presents LATENTGUARD, a latent world logic repair framework for safeguarding AD-VLMs. Instead of relying on explicit rule enumeration or post-hoc output correction, LATENTGUARD learns safety repair from feature-layer changes induced by logic-enhanced safe correction. Unsafe outputs and counterfactual safe alternatives define logic-induced representation shifts, which supervise a conditional latent module to regenerate hidden-state repair signals. Future-consistency regularization further encourages repaired representations to align with safer future evolution during training. Experiments on VRU-Accident and NAVSIM show that LATENTGUARD improves both accident-scene safety and common-scene driving utility, while closed-loop and human-in-the-loop case studies further validate its effect in interactive driving scenarios.

Acknowledgements

This work is supported by Beijing Natural Science Foundation (Grant. QY25225 and Grant. QY25229).

References

[Assran *et al.*, 2023] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *CVPR*, pages 15619–15629, 2023.

[Bardes *et al.*, 2024] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024.

[Brohan *et al.*, 2023] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[Chen *et al.*, 2025] Ruolin Chen, Yinqian Sun, Jihang Wang, Mingyang Lv, Qian Zhang, and Yi Zeng. SafeMind: Benchmarking and mitigating safety risks in embodied LLM agents. *arXiv preprint arXiv:2509.25885*, 2025.

[Choi *et al.*, 2026] Dongwook Choi, Taeyoon Kwon, Bogyung Jeong, Minju Kim, Yeonjun Hwang, Hyojun Kim, Byungchul Kim, Young Kyun Jang, and Jinyoung Yeo. EMBGuard: Constructing hazard-aware guardrails for safe planning in embodied agents. *arXiv preprint arXiv:2605.30924*, 2026.

[Dauner *et al.*, 2024] Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, et al. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. *NeurIPS*, 2024.

[Gokhale *et al.*, 2025] Anand Gokhale, Vaibhav Srivastava, and Francesco Bullo. LogicGuard: Improving embodied LLM agents through temporal logic based critics. *arXiv preprint arXiv:2507.03293*, 2025.

[Gopalkrishnan *et al.*, 2024] Akshay Gopalkrishnan, Ross Greer, and Mohan Trivedi. Multi-frame, lightweight & efficient vision-language models for question answering in autonomous driving. *arXiv preprint arXiv:2403.19838*, 2024.

[Hu *et al.*, 2022] Anthony Hu, Gianluca Corrado, Nicolas Griffiths, Zak Murez, Corina Gurau, Hudson Yeo, Alex Kendall, Roberto Cipolla, and Jamie Shotton. Model-based imitation learning for urban driving. In *NeurIPS*, 2022.

[Hu *et al.*, 2023] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. GAIA-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

[Jia *et al.*, 2026] Feiyang Jia, Lin Liu, Ziying Song, Caiyan Jia, Hangjun Ye, Xiaoshuai Hao, and Long Chen. DriveWorld-VLA: Unified latent-space world modeling with vision-language-action for autonomous driving. *arXiv preprint arXiv:2602.06521*, 2026.

[Kim *et al.*, 2024] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[Kim *et al.*, 2025] Younggun Kim, Ahmed S Abdelrahman, and Mohamed Abdel-Aty. Vru-accident: A vision-language benchmark for video question answering and dense captioning for accident scene understanding. In *ICCV Workshop*, 2025.

[LeCun, 2022] Yann LeCun. A path towards autonomous machine intelligence. *OpenReview*, 2022. Version 0.9.2, 2022-06-27.

[Lu *et al.*, 2026] Xiaoya Lu, Yijin Zhou, Zeren Chen, Ruocheng Wang, Bingrui Sima, Enshen Zhou, Lu Sheng, Dongrui Liu, and Jing Shao. HomeGuard: VLM-based embodied safeguard for identifying contextual risk in household task. *arXiv preprint arXiv:2603.14367*, 2026.

[Ma *et al.*, 2024] Yingzi Ma, Yulong Cao, Jiachen Sun, Marco Pavone, and Chaowei Xiao. Dolphins: Multimodal language model for driving. In *ECCV*, 2024.

[Mao *et al.*, 2023] Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. A language agent for autonomous driving. *arXiv preprint arXiv:2311.10813*, 2023.

[Meng *et al.*, 2022] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *NeurIPS*, 2022.

[Qin *et al.*, 2025] Yiran Qin, Li Kang, Xiufeng Song, Zhenfei Yin, Xiaohong Liu, Xihui Liu, Ruimao Zhang, and Lei Bai. Robofactory: Exploring embodied agent collaboration with compositional constraints. In *ICCV Workshop*, 2025.

[Ravichandran *et al.*, 2025] Zachary Ravichandran, Alexander Robey, Vijay Kumar, George J. Pappas, and Hamed Hassani. Safety Guardrails for LLM-Enabled Robots. *arXiv preprint arXiv:2503.07885*, 2025.

[Sato *et al.*, 2021] Takami Sato, Junjie Shen, Ningfei Wang, Yunhan Jia, Xue Lin, and Qi Alfred Chen. Dirty road can attack: Security of deep learning based automated lane centering under {Physical-World} attack. In *USENIX Security*, 2021.

[Shi *et al.*, 2025] Chen Shi, Shaoshuai Shi, Kehua Sheng, Bo Zhang, and Li Jiang. DriveX: Omni scene modeling for learning generalizable world knowledge in autonomous driving. *arXiv preprint arXiv:2505.19239*, 2025.

[Sima *et al.*, 2024] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li.

DriveIm: Driving with graph visual question answering. In *ECCV*, 2024.

[Tian *et al.*, 2024] Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. DriveVLM: The convergence of autonomous driving and large vision-language models. *arXiv preprint arXiv:2402.12289*, 2024.

[Turner *et al.*, 2023] Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.

[Wang *et al.*, 2024] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. DriveDreamer: Towards real-world-driven world models for autonomous driving. In *ECCV*, 2024.

[Wang *et al.*, 2025] Le Wang, Zonghao Ying, Xiao Yang, Quanchen Zou, Zhenfei Yin, Tianlin Li, Jian Yang, Yaodong Yang, Aishan Liu, and Xianglong Liu. RoboSafe: Safeguarding embodied agents via executable safety logic. *arXiv preprint arXiv:2512.21220*, 2025.

[Wang *et al.*, 2026] Guoqing Wang, Pin Tang, Xiangxuan Ren, Guodongfang Zhao, Bailan Feng, and Chao Ma. Learning vision-language-action world models for autonomous driving. *arXiv preprint arXiv:2604.09059*, 2026.

[Wu *et al.*, 2024] Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. ReFT: Representation finetuning for language models. *arXiv preprint arXiv:2404.03592*, 2024.

[Xiang *et al.*, 2024] Zhen Xiang, Linzhi Zheng, Yanjie Li, Junyuan Hong, Qinbin Li, Han Xie, Jiawei Zhang, Zidi Xiong, Chulin Xie, Carl Yang, Dawn Song, and Bo Li. GuardAgent: Safeguard LLM agents by a guard agent via knowledge-enabled reasoning. *arXiv preprint arXiv:2406.09187*, 2024.

[Zhang *et al.*, 2025a] Jiawei Zhang, Xuan Yang, Taiqi Wang, Yu Yao, Aleksandr Petiushko, and Bo Li. SafeAuto: Knowledge-enhanced safe autonomous driving with multimodal foundation models. *arXiv preprint arXiv:2503.00211*, 2025.

[Zhang *et al.*, 2025b] Tianyuan Zhang, Ting Jin, Lu Wang, Jiangfan Liu, Siyuan Liang, Mingchuan Zhang, Aishan Liu, and Xianglong Liu. Bench2advlm: a closed-loop benchmark for vision-language models in autonomous driving. *arXiv preprint arXiv:2508.02028*, 2025.

[Zhang *et al.*, 2026] Tianyuan Zhang, Peng Yue, Zihao Peng, Jiangfan Liu, Zonghao Ying, Jiakai Wang, Tianlin Li, Jian Yang, Yaodong Yang, Aishan Liu, and Xianglong Liu. GuardAD: Safeguarding autonomous driving MLLMs via markovian safety logic. *arXiv preprint arXiv:2605.10386*, 2026.

[Zhao *et al.*, 2025] Guosheng Zhao, Xiaofeng Wang, Zheng Zhu, Xinze Chen, Guan Huang, Xiaoyi Bao, and Xingang Wang. DriveDreamer-2: LLM-enhanced world models for diverse driving video generation. In *AAAI*, 2025.

[Zheng *et al.*, 2024] Wenzhao Zheng, Weiliang Chen, Yuanhui Huang, Borui Zhang, Yueqi Duan, and Jiwen Lu. OccWorld: Learning a 3D occupancy world model for autonomous driving. In *ECCV*, 2024.

[Zheng *et al.*, 2025] Yupeng Zheng, Pengxuan Yang, Zebin Xing, Qichao Zhang, Yuhang Zheng, Yinfeng Gao, Pengfei Li, Teng Zhang, Zhongpu Xia, Peng Jia, and Dongbin Zhao. World4Drive: End-to-end autonomous driving via intention-aware physical latent world model. *arXiv preprint arXiv:2507.00603*, 2025.

[Zhou *et al.*, 2025] Enshen Zhou, Qi Su, Cheng Chi, Zhizheng Zhang, Zhongyuan Wang, Tiejun Huang, Lu Sheng, and He Wang. Code-as-Monitor: Constraint-aware visual programming for reactive and proactive robotic failure detection. In *CVPR*, 2025.

[Zou *et al.*, 2023] Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023.