

CornerBench: Is Your Driving Model Safe Enough?

Ziqi Shi¹ Yuxue Yang¹ Yanjia Li¹ Maoyuan Shao¹ Yida Wang²
 Xueyang Zhang² Kun Zhan² Lue Fan¹✉ † Zhaoxiang Zhang¹✉

¹Institute of Automation, Chinese Academy of Sciences ²Li Auto Inc.

✉ Corresponding Authors. † Project Lead.

{shiziqi2025, lue.fan, zhaoxiang.zhang}@ia.ac.cn

Abstract

Modern end-to-end driving planners have improved rapidly on common driving benchmarks, but whether they can safely handle rare, high-consequence corner cases remains unclear. Evaluating such cases is difficult because natural driving logs contain few safety-critical events, while staged data collection is costly and difficult to scale. We present **CornerFactory**, an automated pipeline for constructing corner cases from real-world driving logs according to structured scene specifications. CornerFactory supports multi-view-consistent object insertion, novel-view synthesis from perturbed ego poses, and transfer across vehicle platforms with different camera configurations. Building on this pipeline, we introduce **CornerBench**, a benchmark that organizes generated corner cases into a structured taxonomy and evaluates planners using continuous, severity-aware metrics that capture graded safety differences. Using CornerBench, we evaluate representative planners along several dimensions commonly expected to improve robustness, including training data volume and model size, multimodal sensing, language-based scene understanding and chain-of-thought reasoning in VLA planners, and the use of world models for future prediction and planning.

Project page <https://cornerbench.github.io/>

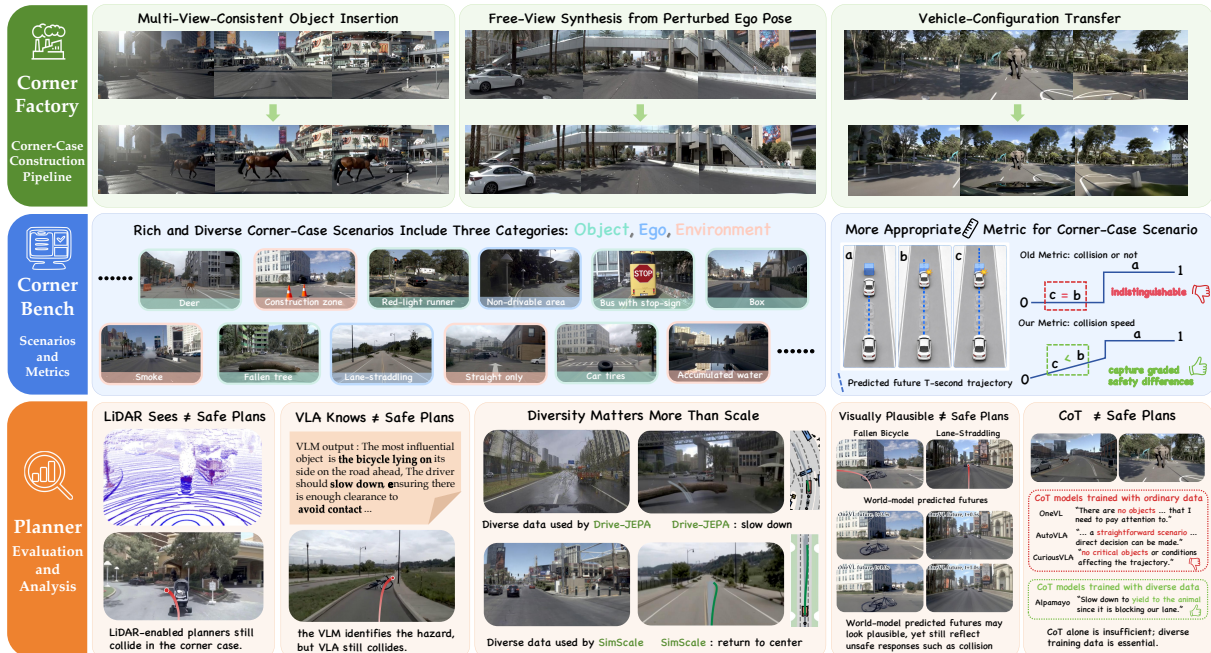


Figure 1: Overview of this paper.

1 Introduction

End-to-end autonomous driving has advanced rapidly in recent years. This paradigm can produce driving behavior that is smoother, more context-aware, and closer to human-like decision making, and it has raised the performance ceiling of autonomous driving systems in many common scenarios.

Existing planning benchmarks have substantially standardized the evaluation of end-to-end driving planners. For example, nuPlan [1], NAVSIM [2], and Bench2Drive [3] provide common testbeds for comparing planners. However, many of these benchmark scenes still follow the distribution of regular driving scenes. This makes it difficult to precisely evaluate how planning models behave in long-tail corner cases, for example when a horse is crossing the road and is about to collide with the ego vehicle. This limitation also makes it harder to design, diagnose, and compare model architectures that are specifically intended to improve corner-case robustness, such as VLA-based planners [4, 5, 6, 7, 8, 9, 10, 11] and world-model-based planners [5, 6, 7, 11, 12, 13, 14, 15].

The lack of corner-case benchmarks is largely caused by the difficulty of obtaining such data. Natural road collection is inefficient because safety-critical long-tail events occur with low probability, while deliberately staging them in the physical world requires high economic cost, safety control, repeated setup, and careful manual inspection. Simulation platforms such as CARLA [16, 17] can create risky situations more easily, but simulated observations may differ from real sensor data and may not match the input interface expected by planners trained on real driving logs. This gap suggests that generative models are a promising solution: they can synthesize realistic driving scenes, reduce the cost of data collection, and make it easier to create targeted corner cases. Recent advances in controllable driving-scene generation and editing make this direction increasingly feasible. Nevertheless, existing methods primarily optimize visual realism and controllability under nominal driving conditions, rather than the targeted construction of rare and safety-critical events. Many methods can already synthesize plausible multi-view observations for routine driving scenarios [18, 19, 20, 21, 22, 23, 24, 25], but only a small body of work explicitly targets corner-case generation [26]. Such risk-oriented generation remains largely limited to a single view, preventing direct evaluation of planners that require multi-view camera inputs.

To close this gap, this report first introduces **CornerFactory**, a pipeline for constructing corner cases with multi-view camera observations from real driving logs. CornerFactory uses a structured scene specification to represent the target case and retrieve a compatible base scene from real driving logs. The pipeline supports multi-view-consistent object insertion for object-centered cases and free-view synthesis from perturbed ego poses for ego-state cases, and can further transfer generated cases to different vehicle configurations so that the same corner case can be used to evaluate planners designed for different vehicle platforms and camera layouts. It also updates LiDAR observations for planners that require LiDAR input and applies VLM-assisted plus human quality filtering before exporting benchmark-ready records. Table 1 compares CornerFactory with representative controllable driving-scene generation and editing methods.

Capability	Magic Drive [18]	Driving Diffusion [19]	Scene Crafter [20]	World Splat [21]	Dream 4Drive [22]	Horizon Forge [27]	Compo SIA [26]	Fail 2Drive [17]	Uni Scene [23]	Cosmos Drive Dreams [24]	X World [25]	Xiaomi Auto WM [28]	GAIA 3 [29]	Waymo WM [30]	Corner Factory
Real-log grounding	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓
Corner-case focus	✗	✗	✗	✗	✗	✗	✓	✓	✗	✗	✗	✓	✓	✓	✓
Multi-camera observations	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓
Cross-vehicle transfer	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✓	✗	✓
LiDAR observations	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✗	✗	✗	✓	✓
Open source	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓

Table 1: Comparison with representative controllable driving generation and editing methods. Rows denote capabilities and columns denote methods.

Building on this construction pipeline, we instantiate **CornerBench**. CornerBench provides a comprehensive taxonomy of generated corner cases, covering object-centric, ego-state, and environmental challenges such as rare obstacles, abnormal road users, off-nominal ego starts, visual cue conflicts, and local environmental hazards. CornerBench further introduces continuous, corner-case-aware metrics that preserve meaningful differences among unsafe behaviors—for example, distinguishing a low-speed collision from a high-speed collision—rather than treating all failed cases identically. Because the scenarios are exported in a NAVSIM-compatible format, the benchmark can be used with a broad set of existing end-to-end planners while preserving a shared evaluation interface. Table 2 compares CornerBench with representative autonomous-driving benchmarks.

Capability	NAVSIM [2]	Bench2Drive [3]	Fail2Drive [17]	WOD-E2E [31]	KITScenes [32]	nuReasoning [33]	CornerBench
Real-log grounding	✓	✗	✗	✓	✓	✓	✓
Corner-case focus	✗	✗	✓	✓	✓	✓	✓
Severity-aware metrics	✗	✗	✗	✗	✗	✗	✓
Economical	✗	✗	✗	✗	✗	✗	✓

Table 2: Comparison with representative autonomous-driving benchmarks. Real-log grounding indicates whether scenarios are grounded in real-world driving records; corner-case focus indicates whether long-tail safety-critical cases are the benchmark’s primary target; severity-aware metrics indicate whether failed trajectories remain distinguishable through graded scores based on violation severity rather than receiving a shared binary outcome; economical indicates whether the benchmark avoids expensive simulation or new data collection.

Building on CornerBench, we investigate which commonly assumed advances actually improve the robustness of modern end-to-end driving models under generated corner cases. Our evaluation is organized around five factors: (1) training data and model scaling, examining how data volume, data diversity, and model size relate to corner-case robustness; (2) multimodal sensing, assessing whether LiDAR enables stronger responses to rare hazards; (3) VLM-based scene understanding, testing whether semantic understanding translates into safer trajectories; (4) CoT reasoning, examining whether its benefits generalize from routine driving scenes to corner cases; and (5) world modeling, evaluating whether predicted futures support appropriate safety-critical responses. Figure 1 summarizes the overall scope of this paper.

In summary, this work makes three main contributions:

- We introduce **CornerFactory**, an automated pipeline for constructing corner cases from real-world driving logs according to structured scene specifications. It supports multi-view-consistent object insertion, novel-view synthesis from perturbed ego poses, transfer across vehicle configurations, and corresponding LiDAR updates for multimodal planners.
- We build **CornerBench**, a NAVSIM-compatible benchmark covering object-centric, ego-state, and environmental corner cases. It combines a structured scenario taxonomy with continuous, severity-aware metrics that distinguish graded unsafe behaviors rather than treating all failed cases identically.
- We systematically evaluate representative end-to-end driving planners on CornerBench to assess their robustness to generated corner cases. Our analysis covers training data scale and diversity, model size, multimodal sensing, VLA reasoning, world models, and recovery from off-nominal ego states.

2 Related Work

Driving Scene Generation and Editing. Recent controllable driving generation methods synthesize realistic driving scenes from structured conditions, while editing methods modify existing driving clips. MagicDrive [18], DrivingDiffusion [19], DriveDreamer [34], GAIA-1 [35], Drive-WM [36], Panacea [37], GenAD [38], Vista [39], DriveDreamer-2 [40], CogDriving [41], UniScene [42], and Genesis [43] use layout, 3D, agent, or action conditions for controllable driving-scene generation. SceneCrafter [20], Dream4Drive [22], and HorizonForge [27] further support driving-scene editing, such as inserting or removing vehicles in existing scenes. WorldSplat [21], MagicDrive3D [44], FreeVS [45], DriveDreamer4D [46], FreeSim [47], StreetCrafter [48], and FlexDrive [49] improve 3D or free-view driving-scene generation and rendering. These methods mainly focus on generating plausible ordinary driving scenes or controllable traffic variations. Risk-oriented methods such as CompoSIA [26] and RiskMV [50] move closer to safety-critical scenario generation, but most existing methods do not directly provide multi-camera observations with cross-view consistency for multi-view planners or the corresponding LiDAR sensor records for multimodal-input planners.

Driving Scene Benchmarks. Driving scene benchmarks establish standardized data, protocols, and metrics for evaluating autonomous-driving planners. nuPlan [1] provides large-scale real-world driving logs, while NAVSIM [2] builds on these logs with standardized routes and planning metrics for end-to-end evaluation. Bench2Drive [3] complements real-log evaluation with closed-loop testing in simulation. Although these general-purpose benchmarks are effective for measuring performance under the nominal driving distribution, they offer limited coverage of rare, safety-critical events. Recent efforts address this gap from two complementary perspectives: Fail2Drive [17] constructs failure-oriented scenarios for closed-loop evaluation, whereas WOD-E2E [31], KITScenes [32], and nuReasoning [33] collect or annotate long-tail cases in real-world driving data, including scenarios that require explicit reasoning. However, the former depends on predefined simulation assets and the latter on rare events present in existing logs; neither provides an economical and flexible mechanism for creating user-specified corner cases while retaining real-log context.

End-to-End Autonomous Driving Planners. Early planning-oriented methods, such as UniAD [51] and VAD [52], established the modern end-to-end paradigm by jointly learning scene representations and future trajectories. Subsequent planners explore several complementary directions that are commonly expected to improve robustness in corner cases. First, VLA planners incorporate VLM-based scene understanding, language-conditioned actions, or intermediate reasoning, as exemplified by ReCogDrive [53], DriveVLA-W0 [5], OneVL [7], DynVLA [6], AutoVLA [10], CuriousVLA [8], and Alpamayo [4]. Second, world-model planners introduce future-representation learning, predictive video objectives, or video-pretrained features to support planning, including DriveLaW [13], WoTE [15], SeerDrive [14], DriveWorld-VLA [11], and Drive-JEPA [12]. Third, sensor-rich planners, such as TransFuser [54], GoalFlow [55], Mimir [56], DiffusionDrive [57], and DiffusionDriveV2 [58], incorporate LiDAR to provide explicit geometric information. Finally, methods such as SimScale [59], Drive-JEPA [12], and RAP [60] exploit additional training data or simulated experience to improve generalization. These directions are not mutually exclusive, but together represent the principal design choices currently used to strengthen end-to-end driving planners.

3 CornerFactory

CornerFactory is an automated pipeline for constructing corner cases from real-world driving logs. The pipeline begins with a structured scene specification that defines the desired corner case and guides the retrieval of a compatible base scene from real-world driving logs. From the retrieved base scene, the pipeline constructs object-centered cases through multi-view-consistent object insertion and ego-state cases through free-view synthesis from perturbed ego poses. It can further transfer generated cases to different vehicle configurations, allowing the same corner case to be used to evaluate planners across vehicle platforms and camera layouts. It then updates the LiDAR observations to match the generated camera views so that the generated cases can also be used to evaluate planners that require LiDAR input. Finally, a VLM-based reviewer screens the generated clips for mismatch with the scene specification, temporal inconsistency, and obvious visual artifacts before human filtering.

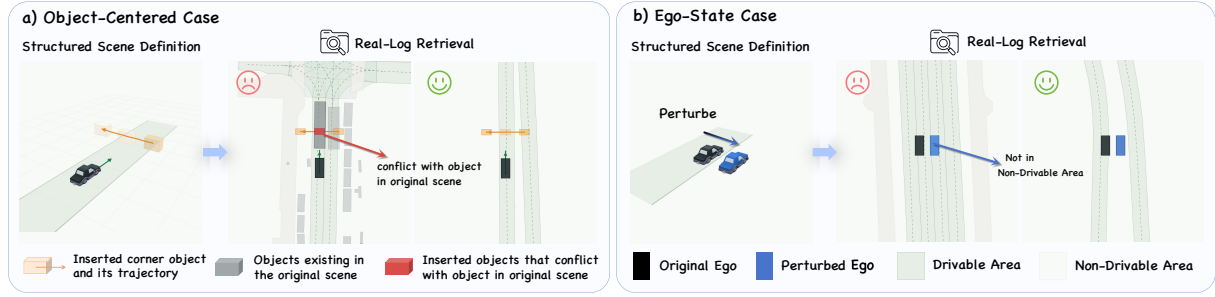


Figure 2: **Structured scene specification and real-log retrieval for object-centered and ego-state cases.** The specification defines the intended object hazard or ego-state perturbation, while retrieval selects a compatible base scene from real driving logs.

3.1 Structured Scene Specification and Real-Log Retrieval

This stage determines both what corner case to construct and where it can be plausibly instantiated. The structured scene specification represents the intended intervention with explicit 3D spatial and temporal attributes, while real-log retrieval uses these attributes as constraints to select a compatible base scene from real driving logs. We consider two categories of specifications, object-centered hazards and ego-state perturbations, as illustrated in Figure 2.

For object-centered cases, the specification defines an inserted object with a semantic type that is rare in autonomous-driving scenes, a plausible 3D box size, and a location and temporal motion chosen to intersect the original ego trajectory in the future and thereby create a targeted hazard for evaluating whether a planner can slow down, stop, or steer away safely. Real-log retrieval then selects suitable base scenes from real driving logs that satisfy compatibility constraints, such as avoiding collisions between the inserted object’s specified trajectory and existing non-ego agents, to keep the insertion physically plausible in the generated scene.

For ego-state cases, the specification defines an off-nominal ego pose, such as a lane-straddling or non-drivable-area start, to require recovery toward a nominal driving state and thereby create a targeted case for evaluating whether a planner can recover toward the lane center rather than simply continuing from the perturbed ego state. Real-log retrieval then selects suitable base scenes from real driving logs that satisfy compatibility constraints, such as road geometry and surrounding traffic that can support the specified perturbation.

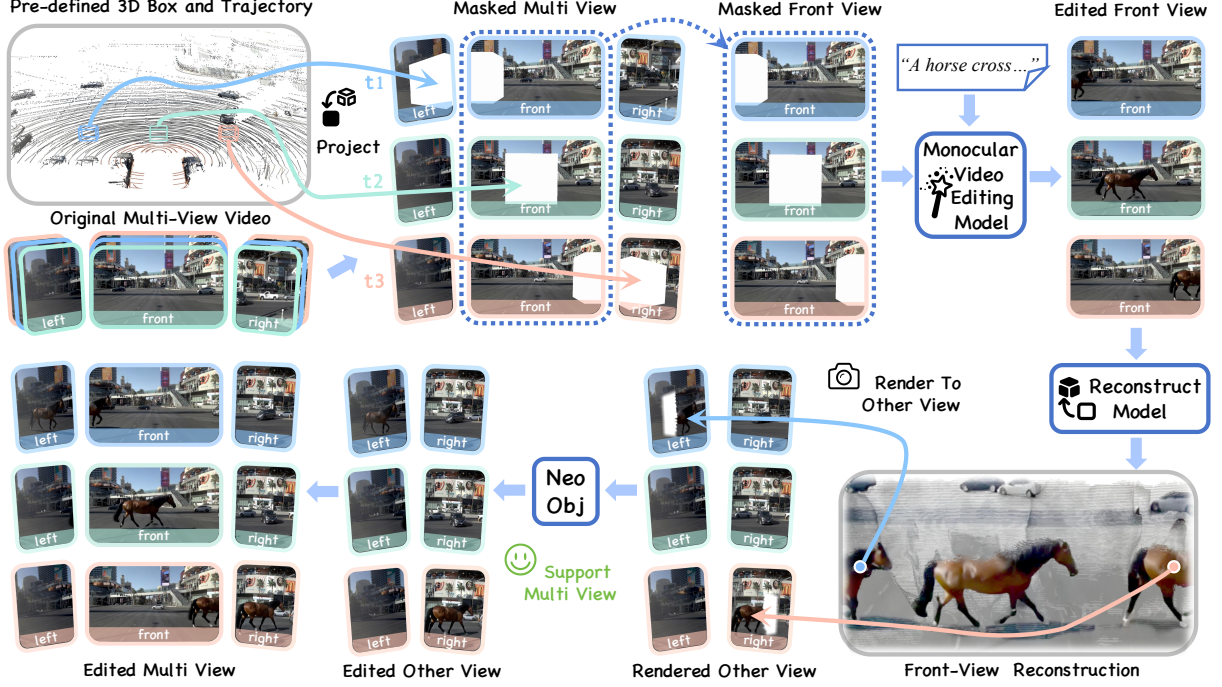


Figure 3: **Multi-view-consistent object insertion.** A controllable video editing model first inserts the corner object into the front-view video. The edited front view is then reconstructed and rendered into the remaining camera views, where NeoObj completes the target-view appearance.

3.2 Multi-View-Consistent Object Insertion

Following the structured scene specification, CornerFactory performs multi-view-consistent insertion of the specified object into the retrieved base scene. Figure 3 illustrates this object insertion pipeline. Given the 3D object specification, the pipeline first projects the box trajectory into all camera views to obtain temporally and cross-view consistent object-insertion masks. These projections turn the 3D placement into explicit 2D image-space constraints, allowing the corner object to be inserted at the intended position rather than generated freely. The masked front-view video and a semantic prompt describing the target object and its behavior are then provided to a controllable video editing model; in our implementation we use VACE [61] to synthesize the visual appearance of the inserted corner object while preserving the original surrounding scene content. However, VACE performs single-view video editing and does not produce corresponding observations of the inserted object in the other camera views. This reflects a broader limitation of current video editing models, which generally lack support for jointly editing synchronized multi-camera views.

To overcome this single-view limitation, we introduce a pipeline that transfers the inserted object from the edited front view to the remaining camera views. The pipeline first reconstructs the edited front-view scene with a feed-forward reconstruction model. In our implementation, we use Depth Anything 3 (DA3) [62] to recover 3D geometry from the edited front-view video and our fine-tuned Gaussian head to produce per-frame Gaussian representations. The reconstructed content is then rendered into the precomputed masked regions of each target view as geometric and appearance conditions, after which NeoObj refines the transferred object and completes its appearance in the target views. Its training details are provided in Appendix A.1. This design keeps the inserted object tied to a specific 3D trajectory while producing photorealistic observations in which its geometry and appearance remain consistent across camera viewpoints.

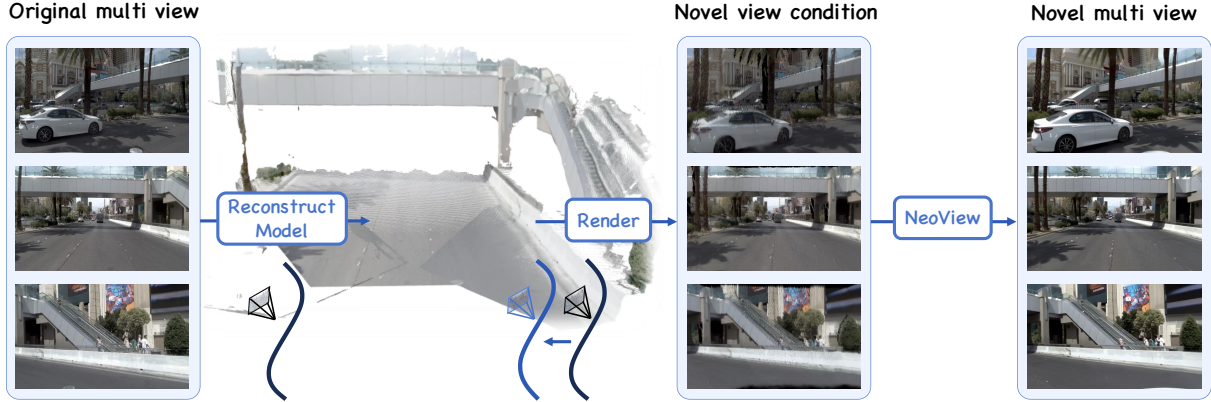


Figure 4: **Free-view synthesis from perturbed ego poses.** The original scene is first reconstructed. It is then rendered from the perturbed ego poses defined in the scene specification to produce novel-view conditions, after which NeoView completes newly visible regions.

3.3 Free-View Synthesis from Perturbed Ego Poses

Ego-state corner cases perturb the ego vehicle’s pose while preserving the underlying map, route, and surrounding traffic. CornerFactory constructs these cases through free-view synthesis, generating camera observations from the perturbed ego poses defined in the structured scene specification. Figure 4 illustrates this pipeline. The pipeline reuses the DA3-based reconstruction stack from the object-insertion pipeline to obtain per-frame Gaussian representations of the original multi-view scene. It then renders these representations along the perturbed ego trajectory to produce novel-view conditions, after which NeoView synthesizes the newly visible content. Its training details are provided in Appendix A.2.

By perturbing only the ego trajectory, the pipeline synthesizes new camera viewpoints while preserving the original map, route, and surrounding traffic. This makes it possible to test whether a planner simply extrapolates the perturbed initial state or actively recovers toward a safer nominal driving pose.

3.4 Vehicle-Configuration Transfer

Beyond constructing corner cases for the source vehicle, CornerFactory can transfer each generated case to a target vehicle configuration while preserving the underlying scene. Figure 5 illustrates this vehicle-configuration transfer pipeline. This capability enables the same corner case to be rendered under different camera configurations, allowing planners designed for different vehicle platforms to be evaluated on equivalent scenarios. The pipeline reuses the DA3-based reconstruction stack from the object-insertion pipeline to obtain per-frame Gaussian representations of the source scene. It then re-renders these representations using the camera parameters of the target vehicle configuration to produce a novel-embodiment condition, after which NeoEmb completes the newly exposed regions to synthesize the target embodiment. Its training details are provided in Appendix A.3.

The resulting observations retain the original corner-case geometry and traffic context but match the visual input format expected by the target planner. This capability allows corner cases constructed from nuPlan scenes to evaluate planners trained with other camera configurations, such as NVIDIA’s Alpamayo family.

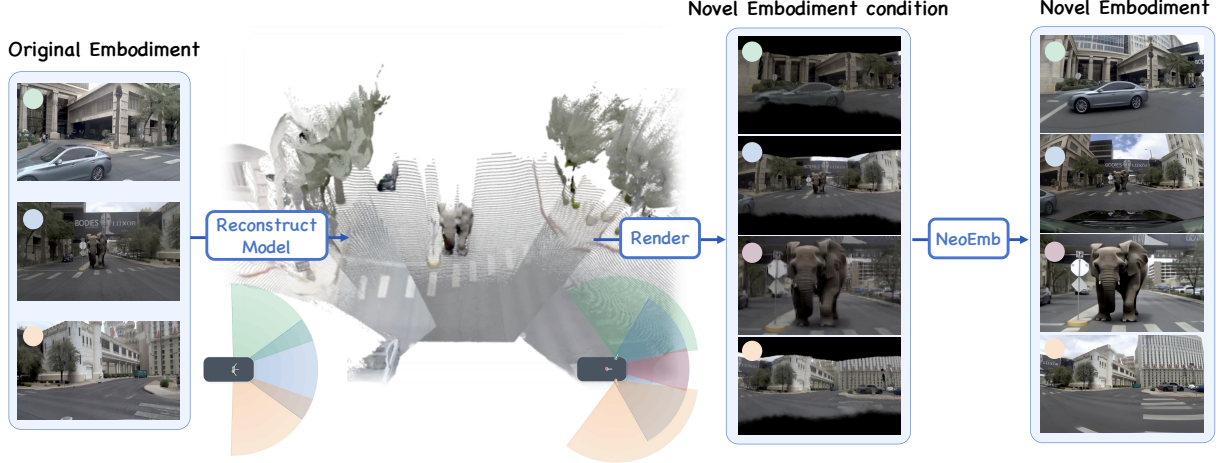


Figure 5: **Vehicle-configuration transfer.** The source scene is first reconstructed and then rendered using the target camera parameters to produce a novel-embodiment condition, after which NeoEmb completes newly exposed regions to synthesize the target embodiment.

3.5 Aligning LiDAR with Edited Camera Views

Camera-based editing alone is insufficient for multimodal planners because an inserted object would appear in the RGB observations but remain absent from the original LiDAR point cloud. To restore cross-modal consistency, CornerFactory uses the same feed-forward reconstruction pipeline to reconstruct the edited multi-view observations as 3D Gaussians and identifies the Gaussians associated with the inserted content. We then employ Gaussian-based ray tracing [63] to synthesize the corresponding LiDAR returns. The synthesized returns replace the inconsistent original measurements, yielding geometrically aligned camera and LiDAR observations. The complete alignment pipeline is illustrated in Figure 6.

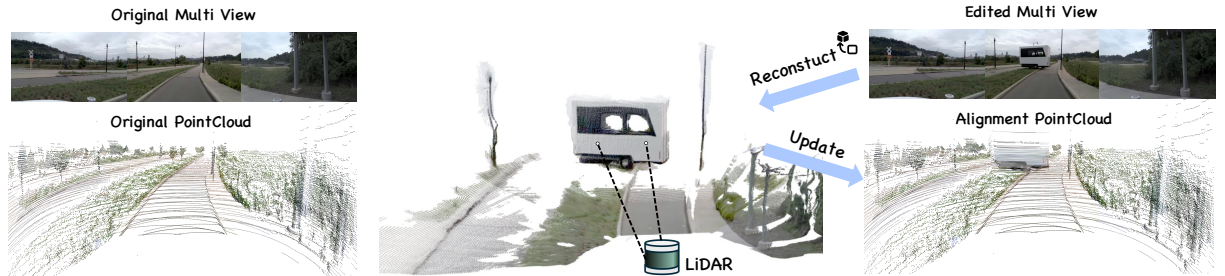


Figure 6: **LiDAR alignment for edited corner cases.** CornerFactory uses feed-forward reconstruction to obtain 3D Gaussians from the edited multi-view observations and synthesizes the corresponding LiDAR returns, producing geometrically aligned camera and LiDAR observations.

4 CornerBench

Using the CornerFactory construction pipeline described above, we build CornerBench as a suite of generated driving scenarios that stress end-to-end planners. Each benchmark case starts from a real driving clip and uses the pipeline to produce NAVSIM-compatible scene records, including multi-view camera observations and aligned LiDAR observations. To capture meaningful differences among planner responses, CornerBench evaluates predicted trajectories using continuous, severity-aware metrics.

4.1 Scenario Taxonomy

Using the pipeline described above, we construct a suite of meaningful corner cases that are challenging for driving and informative for evaluating current planning models. CornerBench organizes generated cases into three major groups: **Ego-state** cases, such as a lane-straddling start; **Environmental** cases, such as smoke; and **Object-centric** cases. Object-centric cases are further divided according to whether the challenge comes from an uncommon object, such as a fallen bicycle; uncommon behavior of a common object, such as a red-light runner; or uncommon texture of a common object, such as a bus with a stop-sign marking. Figure 7 shows this taxonomy with representative examples.

Ego-state. In most real driving logs, the ego vehicle starts from nominal states, such as driving near the lane center. CornerBench creates ego-state corner cases, producing two case types: *Lane-Straddling Start*, where the ego vehicle initially overlaps a lane marking, and *Non-Drivable-Area Start*, where the ego vehicle starts from a non-drivable area. These cases evaluate whether a planner can actively recover toward the lane center or another safer nominal state instead of simply extrapolating the perturbed initial pose.

Object: uncommon objects. This object-centric family inserts rare static or dynamic obstacles near the future ego path. Examples include unusual obstacles, such as *a fallen bicycle, tree branch, baby stroller, and elephant*, as well as rare road users, such as *elephant, horse, and deer*. These cases evaluate whether planners can recognize rare objects and agents, avoid collisions, and yield when needed.

Object: uncommon behavior. CornerBench inserts common traffic participants with abnormal behavior, including path-intrusion cases such as *crossing vehicles, crossing pedestrians, and cut-ins*, where another road user enters the ego lane and creates an imminent collision risk, and other special cases such as *red-light runners*, where another vehicle runs a red light and tests whether the ego planner stops instead of causally following the violating vehicle through the red light. These cases evaluate whether planners can react to interaction-critical motion and rule violations.

Object: uncommon texture. This object-centric family keeps the object category familiar but changes its visual appearance with an unusual cue, such as *a bus carrying a stop-sign marking* or another traffic-sign-like texture. These cases evaluate whether planners can distinguish object identity from misleading local texture.

Environment. This family edits local scene conditions that affect traversability, as well as traffic signs and markings. Examples include *smoke, fire, and construction zones* that affect traversability, and *edited lane-arrow markings* that simulate stale navigation cues. These cases evaluate whether planners respond to scene-level context.

Together, these categories help distinguish whether a planner fails because it cannot recover from an abnormal ego state, fails to recognize rare objects, responds poorly to abnormal agent behavior, misuses visual cues, or misses local environmental hazards.

Following this taxonomy, we construct 230 cases spanning 19 concrete scenario types from 125 source scenes in the nuPlan test set [1]; Appendix B reports the detailed distribution.

4.2 Metrics

CornerBench metrics are designed to expose graded differences among planners. The binary pass/fail criteria used in standard NAVSIM evaluation are often too coarse for corner-case evaluation. Because CornerBench deliberately concentrates on difficult long-tail cases, many planners can receive the same failing outcome under a binary scoring protocol that assigns zero

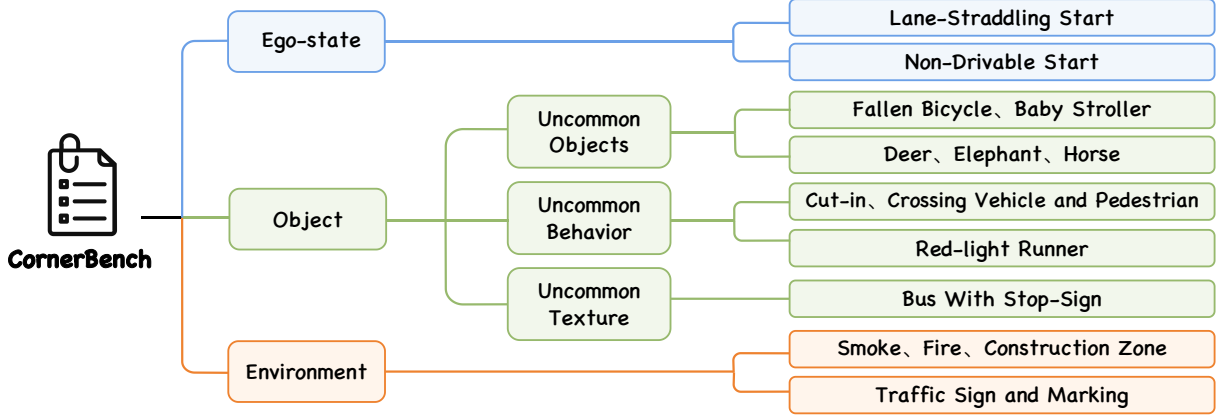


Figure 7: **CornerBench scenario taxonomy.** The suite covers ego-state perturbations, rare or unusual objects, uncommon object behavior, uncommon visual texture, and environmental hazards.

to any failed case, which collapses their differences into a large group of zeros. This makes the metric weakly discriminative precisely in the regime where robustness differences matter most: two planners may both fail, for example by colliding with the hazard, but one may slow down before impact while another drives through the hazard at high speed. We therefore score each case on a continuous 0–1 scale using severity terms. Let $\text{clip}_{[0,1]}(x) = \min(\max(x, 0), 1)$.

Hazard response. For both static and dynamic object cases, let $c \in \{0, 1\}$ denote collision, v_{col} the ego speed at the collision moment, and v_{max} a normalization speed. The score is

$$S_{\text{hazard}} = \begin{cases} 1, & c = 0, \\ 0.6 \left(1 - \text{clip}_{[0,1]} \left(\frac{v_{\text{col}}}{v_{\text{max}}} \right) \right), & c = 1. \end{cases} \quad (1)$$

Thus collision-free behavior receives full credit, while colliding trajectories remain distinguishable by collision speed. This metric is mainly used for *Uncommon Objects*, *Uncommon Behavior*, and the Environment cases that introduce traversability hazards.

Lane keeping. Following the NAVSIM-style lane-keeping rule, let t_{out} be the total duration within the 4-second planning horizon for which the ego trajectory deviates from the lane center by more than 0.5 m. Instead of assigning a binary failure once t_{out} reaches 2 s, CornerBench uses

$$S_{\text{lane}} = \begin{cases} 1, & t_{\text{out}} < 2, \\ 0.6 \left(1 - \text{clip}_{[0,1]} \left(\frac{t_{\text{out}} - 2}{2} \right) \right), & t_{\text{out}} \geq 2. \end{cases} \quad (2)$$

Thus a threshold-level lane-keeping failure receives 0.6, and the score decreases linearly to 0 as the violation duration approaches the full 4-second horizon. This metric is used for ego-state cases, including *Lane-Straddling Start* and *Non-Drivable-Area Start*.

Traffic-rule compliance. For red-light cases, let ρ_{red} be the maximum overlap ratio between the ego footprint and the red-light forbidden polygon over the trajectory, normalized by the ego footprint area. The score is

$$S_{\text{red}} = \begin{cases} 1, & \rho_{\text{red}} = 0, \\ 0.6 \left(1 - \text{clip}_{[0,1]}(\rho_{\text{red}}) \right), & \rho_{\text{red}} > 0. \end{cases} \quad (3)$$

This preserves the NAVSIM-style no-overlap requirement while distinguishing minor boundary intrusions from large overlaps. This metric is used for *red-light runners*.

5 Experiments

We evaluate a broad range of end-to-end planners on CornerBench to assess their robustness under generated corner cases and thereby address the central question of this report: *Are Modern End-to-End Driving Models Safe Enough?* Section 5.1 summarizes the evaluated planners used in our comparisons, while Section 5.2 presents the quantitative results and qualitative failure analyses.

5.1 Evaluated Planners

Our evaluation covers a broad set of end-to-end planners, with particular attention to three design directions often used to improve corner-case robustness: VLA planning with language reasoning, world modeling, and LiDAR-based sensing. We also track whether each planner uses extra data beyond the default training setting to examine whether additional training sources translate into measurable gains.

VLA planners and CoT reasoning. Vision-language-action (VLA) planners, including CoT-enabled variants, use the scene understanding and language reasoning capabilities of VLMs to support action planning. Rather than relying only on visual feature extraction, these methods introduce textual descriptions, intermediate reasoning traces, or language-conditioned action representations so that object relations and traffic context can inform the planner. The evaluated VLA planners, including CoT-enabled variants, are Alpamayo R1 and Alpamayo 1.5 [4], OneVL [7], DynVLA [6], WAM-Flow [9], DriveVLA-W0 [5], AutoVLA [10], CuriousVLA [8], and ReCogDrive-IL/RL with 2B and 8B VLM backbones [53].

World-model planners. World-model planners explicitly learn scene evolution, future representations, future-conditioned scoring signals, or video-pretrained representations that capture driving dynamics for downstream planning. The evaluated planners in this group are Drive-JEPA and Drive-JEPA-PF [12], DriveLaW [13], OneVL [7], DynVLA [6], WoTE [15], SeerDrive [14], DriveVLA-W0 [5], and DriveWorld-VLA [11].

Planners with LiDAR. Camera-plus-LiDAR planners let us test whether additional geometric sensing is sufficient for rare object response. The evaluated LiDAR-enabled planners are WoTE [15], TransFuser [54], SeerDrive [14], GoalFlow [55], Mimir [56], DiffusionDrive [57], and DiffusionDriveV2 [58].

Extra-data planners. We mark a planner as using Extra data when it uses additional driving scenes, video pretraining, or synthetic augmentation beyond the default NAVSIM navtrain setting. Alpamayo R1 and Alpamayo 1.5 [4] use large-scale multi-camera driving data, reported as 80,000 driving hours. SimScale [59] uses synthetic-real co-training with 100K NAVSIM real-world scenarios plus 384K diverse off-nominal scenes synthesized by perturbing ego trajectories in ordinary driving scenarios, comprising 237K planner-based and 147K recovery-based examples. DriveLaW [13] uses nuPlan and nuScenes for latent driving-world/video generation. WAM-Flow [9] reports training on the full 668K-sample nuPlan training set rather than only the 103K NAVSIM navtrain subset. Drive-JEPA and Drive-JEPA-PF [12] use 208 hours of driving-video pretraining. RAP [60] uses scalable 3D rasterization to generate synthetic counterfactual viewpoints, recovery maneuvers, and cross-agent views for planner training. Additional QA or reasoning annotations built on the same NAVSIM scenes, such as those used by ReCogDrive [53] and AutoVLA [10], are treated as annotation enrichment rather than extra driving-scene data in Figure 11.

Table 3: CornerBench planner comparison, sorted in descending order by Hazard-response score from the left block to the right block. Higher is better. VLA denotes vision-language-action; WM, world model; Extra, additional data beyond NAVSIM navtrain; and Sens., sensor type (C: camera, C+L: camera plus LiDAR, N: ego status only). Colored superscripts mark the top three planners for each metric.

Planner	VLA	WM	Extra	Sens.	Hazard response	Lane keeping	Traffic-rule compliance	Planner	VLA	WM	Extra	Sens.	Hazard response	Lane keeping	Traffic-rule compliance
Alpamayo R1 [4]	✓	✗	✓	C	0.476 ¹	0.081	0.740 ³	RAP [60]	✗	✗	✓	C	0.438	0.109	0.016
SimScale [59]	✗	✗	✓	C	0.475 ²	0.274 ¹	0.312	Drive-JEPA-PF [12]	✗	✓	✓	C	0.437	0.060	0.253
Drive-JEPA [12]	✗	✓	✓	C	0.475 ³	0.148 ³	0.084	MeanFuser [68]	✗	✗	✗	C	0.436	0.099	0.283
GTRS-Aug [64]	✗	✗	✗	C	0.468	0.122	0.133	AutoVLA [10]	✓	✗	✗	C	0.436	0.000	0.442
GTRS-Dense [64]	✗	✗	✗	C	0.461	0.146	0.395	Ego-MLP [2]	✗	✗	✗	N	0.436	0.014	0.217
Alpamayo 1.5 [4]	✓	✗	✓	C	0.457	0.143	0.890 ²	GoalFlow [55]	✗	✗	✗	C+L	0.435	0.062	0.182
Hydra-MDP [65]	✗	✗	✗	C	0.454	0.177 ²	0.304	DiffusionDriveV2 [58]	✗	✗	✗	C+L	0.434	0.120	0.062
VeteranAD [66]	✗	✗	✗	C	0.452	0.051	0.128	Mimir [56]	✗	✗	✗	C+L	0.434	0.129	0.198
DriveLaW [13]	✗	✓	✓	C	0.444	0.059	0.292	LEAD [69]	✗	✗	✗	C	0.434	0.071	0.312
DrivoR [67]	✗	✗	✗	C	0.443	0.139	0.181	SparseDriveV2 [70]	✗	✗	✗	C	0.434	0.104	0.040
SeerDrive [14]	✗	✓	✗	C+L	0.441	0.138	0.277	DriveWorld-VLA [11]	✓	✓	✗	C	0.433	0.129	0.284
DriveVLA-W0 [5]	✓	✓	✗	C	0.441	0.059	0.166	DiffusionDrive [57]	✗	✗	✗	C+L	0.432	0.030	0.214
OneVL [7]	✓	✓	✗	C	0.441	0.085	0.506	LTF [2]	✗	✗	✗	C	0.432	0.023	0.337
DynVLA [6]	✓	✓	✗	C	0.441	0.080	0.246	ReCogDrive-IL-8B [53]	✓	✗	✗	C	0.430	0.064	0.361
WoTE [15]	✗	✓	✗	C+L	0.440	0.088	0.228	ReCogDrive-IL-2B [53]	✓	✗	✗	C	0.430	0.098	0.427
CuriousVLA [8]	✓	✗	✗	C	0.440	0.101	0.210	PRIX [71]	✗	✗	✗	C	0.429	0.051	0.279
WAM-Flow [9]	✓	✗	✓	C	0.439	0.094	0.923 ¹	ReCogDrive-RL-8B [53]	✓	✗	✗	C	0.424	0.099	0.038
TransFuser [54]	✗	✗	✗	C+L	0.438	0.046	0.240	ReCogDrive-RL-2B [53]	✓	✗	✗	C	0.424	0.128	0.166

5.2 Results

We organize our quantitative results and qualitative failure analyses around five questions concerning the factors most often expected to improve the corner-case robustness of end-to-end planners:

1. How do training-data scale and model size affect corner-case robustness?
2. How much do multimodal sensors help under rare and safety-critical corner cases?
3. How much does language-based scene understanding help VLA planners under corner-case scenarios?
4. How much does CoT help VLA planners handle corner cases?
5. How much does world model help planners respond under corner-case scenarios?

5.2.1 Analysis of Training-Data Scale and Model Size

Q1. How do training-data scale and model size affect corner-case robustness?

Training-data scale. Alpamayo R1 and Alpamayo 1.5 [4] perform strongly with large-scale external driving data and without using the NAVSIM navtrain set, showing strong cross-domain generalization. This suggests that increasing training-data volume can bring clear gains to planning models. However, data volume alone does not explain the quantitative results. Table 3 summarizes the overall CornerBench comparison using the corner-case-aware metrics defined in Section 4, while Figure 11 relates the Hazard-response scores to training-data volume and model size. Drive-JEPA [12], SimScale [59], DriveLaW [13], and WAM-Flow [9] use additional data at comparable scales. Nevertheless, Drive-JEPA and SimScale achieve Hazard-response scores of 0.475 and 0.475, respectively, compared with 0.444 for DriveLaW and 0.439 for WAM-Flow. A closer inspection of their training data reveals a key difference in data composition. DriveLaW and WAM-Flow rely primarily on standard driving data from nuPlan and nuScenes. By contrast, Drive-JEPA and SimScale expand training-data diversity in different ways: the former uses



Figure 8: **Open-world corner cases from DrivingDojo.** Representative frames from one of the driving-video data sources used by Drive-JEPA cover unusual road obstructions, animal interactions, and nighttime roadside hazards.

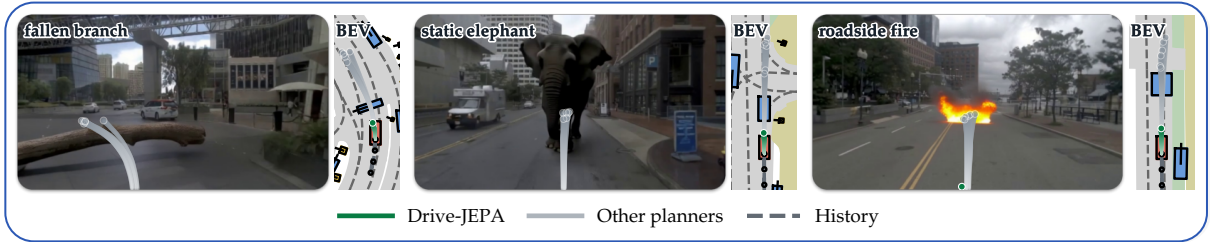


Figure 9: **Drive-JEPA's response behavior in the CornerBench test environment.** Across cases involving moving or stationary objects and unusual environmental hazards, Drive-JEPA produces a more conservative response than other planners. No green trajectory appears in the front-view panels because Drive-JEPA slows down early, making its planned trajectory too short to project into the images.

driving videos covering diverse long-tail events, while the latter perturbs ego trajectories in ordinary driving scenes to synthesize diverse off-nominal situations.

Drive-JEPA pretrains its V-JEPA backbone on 208 hours of driving video assembled from CoVLA [72], DrivingDojo [73], and other datasets before training the downstream NAVSIM planner. This collection exposes the visual encoder to a broader range of object interactions than NAVSIM-style planning data alone. Figure 8 shows representative open-world corner cases from DrivingDojo, including overhanging branches, animals near the ego lane, and nighttime roadside fire. This training-data diversity is reflected in Drive-JEPA's performance in our test environment. Drive-JEPA ranks among the strongest planners on the CornerBench Hazard-response score in Table 3, and Figure 9 shows cases in which it produces a more conservative response to the corner-case hazard than other planners.

SimScale [59] augments its training data with 384K synthetic off-nominal scenes. To construct these scenes, it first perturbs ego trajectories in ordinary driving scenes to create off-nominal ego states, and then uses rule-based strategies or existing planners to generate trajectories that return the vehicle toward normal lane-centered states, providing explicit supervision for ego-state recovery. This targeted training-data augmentation is reflected in SimScale's performance in our test environment. SimScale achieves the highest Lane-keeping score of 0.274 in Table 3, and Figure 10 shows cases in which it steers back toward the lane center from *Lane-Straddling Start* and *Non-Drivable-Area Start*, whereas most planners continue along the perturbed initial pose.

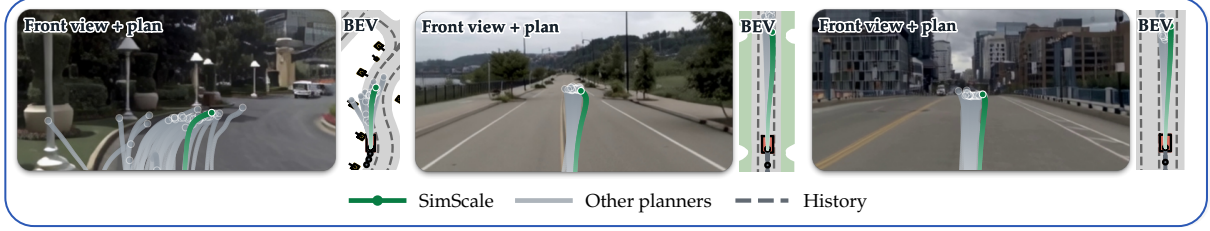


Figure 10: **SimScale’s recovery behavior under Lane-Straddling Start and Non-Drivable-Area Start.** In both types of cases, SimScale actively moves back toward the lane center, while most other planners continue along the perturbed initial state.

For Drive-JEPA, driving-video pretraining on long-tail events is associated with stronger responses to unusual objects. For SimScale, training with synthetic off-nominal states and recovery trajectories is associated with clearer lane-centering behavior from perturbed starts. Together, these cases suggest that diversifying the training data to cover rare objects and off-nominal ego states is more important for corner-case robustness than merely increasing the volume of routine driving data.

Model size. The right panel of Figure 11 shows no clear positive relationship between reported model size and CornerBench Hazard-response score. This is also visible within the same ReCogDrive family [53]: scaling the backbone from 2B to 8B leaves the Hazard-response score unchanged in both settings, at 0.430 for IL and 0.424 for RL, while models with smaller active planners such as Drive-JEPA [12] and SimScale [59] remain among the strongest methods. These comparisons suggest that simply increasing active model size is not sufficient; corner-case robustness also depends on data diversity.

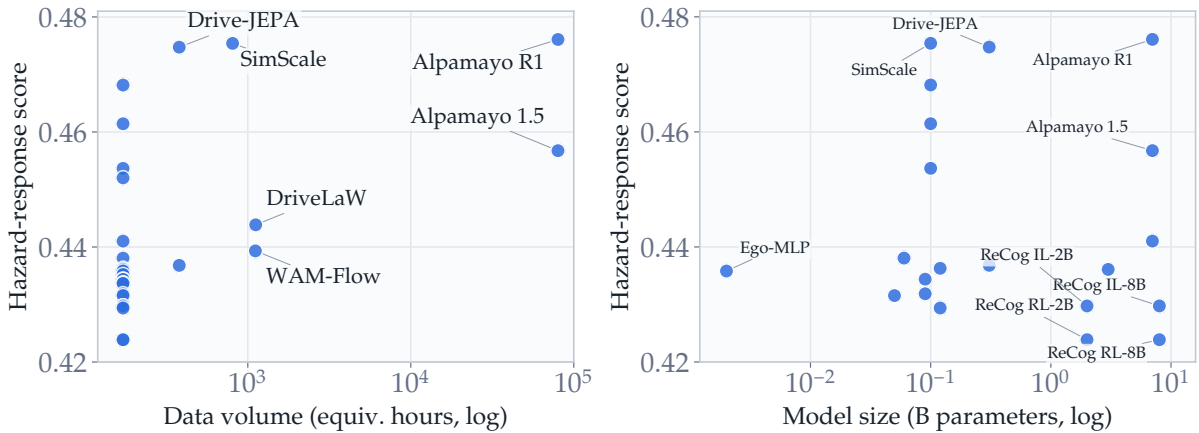


Figure 11: **Scale factors versus CornerBench Hazard-response score.** Left: training-data volume normalized to equivalent driving hours. Right: model size measured in billions of parameters. Both x-axes use logarithmic scale.

A1. Training-data diversity is more important for corner-case robustness than simply increasing data volume or model size.

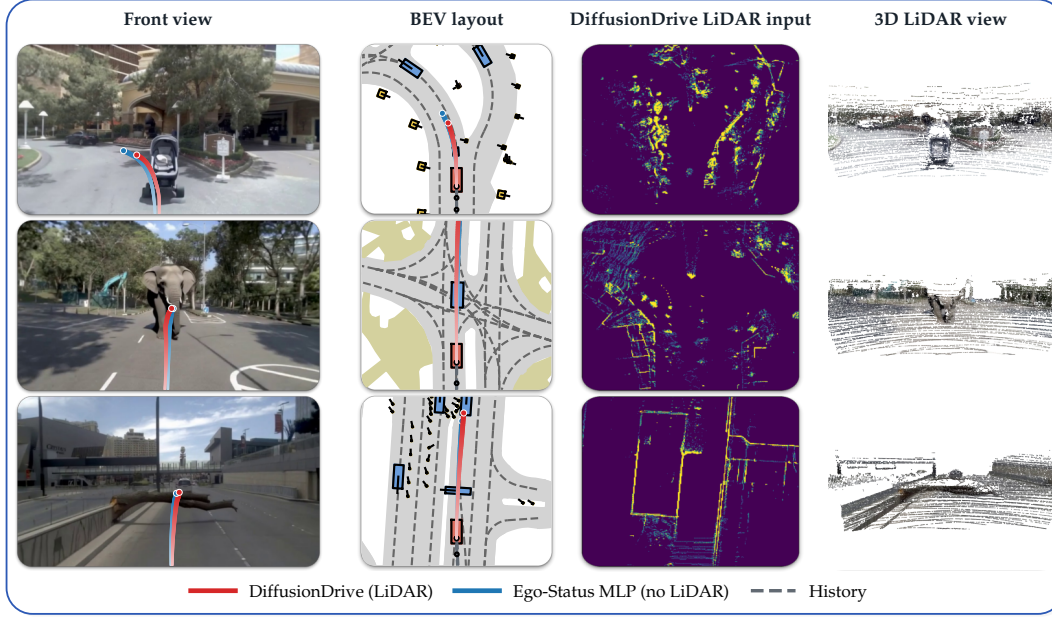


Figure 12: **Planner failures with and without LiDAR.** Across three CornerBench test scenes, DiffusionDrive receives a 2D LiDAR BEV input, whereas Ego-Status MLP uses neither LiDAR nor camera input. Both planners fail to avoid the frontal obstacle, resulting in similarly low Hazard-response scores.

5.2.2 Analysis of Multimodal Sensing

Q2. How much do multimodal sensors help under rare and safety-critical corner cases?

Many evaluated planners use LiDAR together with camera inputs to obtain more reliable geometric perception of object distances and spatial structure [14, 15, 54, 55, 56, 57, 58]. However, the quantitative ranking in Table 3 shows that several top-performing planners do not use a LiDAR sensor. Figure 12 further shows qualitative results on three CornerBench test scenes: DiffusionDrive receives a 2D LiDAR BEV input, whereas Ego-Status MLP uses no LiDAR and does not even use camera input; nevertheless, the trajectories from both planners fail to avoid the frontal obstacle. This indicates that adding a LiDAR sensor does not by itself ensure stronger corner-case response.

A2. Adding a LiDAR sensor does not by itself ensure stronger corner-case response.

5.2.3 Analysis of Language-Based Scene Understanding

Q3. How much does language-based scene understanding help VLA planners under corner-case scenarios?

VLA planners leverage the strong scene-understanding capability of VLMs to incorporate visual observations and language-level representations into action planning [4, 7, 8, 10, 53]. In Figure 13, ReCogDrive’s VLM output identifies the inserted hazard or red traffic light and proposes a reasonable high-level response, but the final trajectory remains inconsistent with that response. For example, in the horse-crossing case, the VLM output first recognizes the horse crossing ahead and then recommends braking to a complete stop, but the planner’s trajectory output still collides with the horse. This shows that VLM scene understanding is not reliably converted into correct trajectory-planning output.

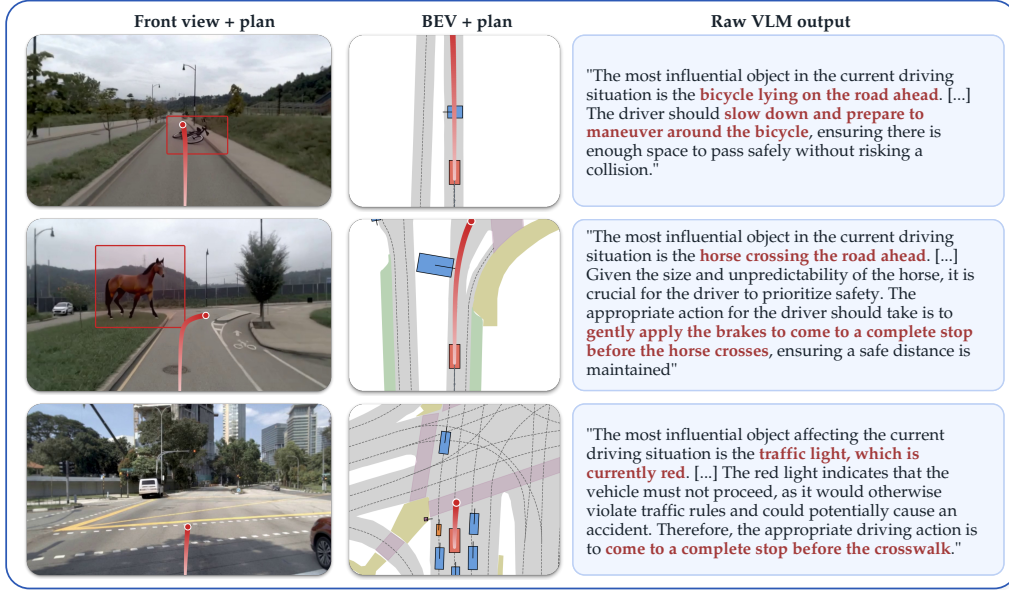


Figure 13: **Examples where VLM scene understanding is not reliably converted into safe trajectories.** Although the VLM output identifies the hazard or red traffic light and suggests a safety-aware response, the planned trajectory still collides with the obstacle or fails to stop at the red light.

A3. VLMs can show strong scene understanding, but VLA planners do not always convert that understanding into safe trajectory plans.

5.2.4 Analysis of Chain-of-Thought Reasoning

Q4. How much does CoT help VLA planners handle corner cases?

CoT is widely regarded as a promising way to enhance the reasoning and planning capabilities of VLA planners in complex driving situations [4, 7, 8, 10]. We test these four CoT-enabled VLA planners on the same corner-case scenes. In Figure 14, OneVL states that there are no relevant objects, CuriousVLA marks the critical object category as absent, and AutoVLA treats the scene as a straightforward driving case. Alpamayo 1.5 instead recognizes the corner-case hazard and suggests a safer response. The first three planners are trained primarily on regular driving scenes, whereas Alpamayo 1.5 uses 80,000 hours of diverse driving data. This comparison suggests that CoT alone does not ensure generalization to corner cases; its effectiveness also depends strongly on the diversity of the training data.

A4. CoT planners trained only on regular driving scenes may fail to generalize to corner cases.

5.2.5 Analysis of World Models

Q5. How much does world model help planners respond under corner-case scenarios?

World models are widely expected to improve driving planning by predicting how a scene may evolve and using the predicted future to inform planning [11, 13, 14, 15]. To examine whether this benefit extends to corner-case scenarios, we inspect the future images decoded by OneVL [7] together with its corresponding planned trajectories on CornerBench. As shown in Figure 15, the

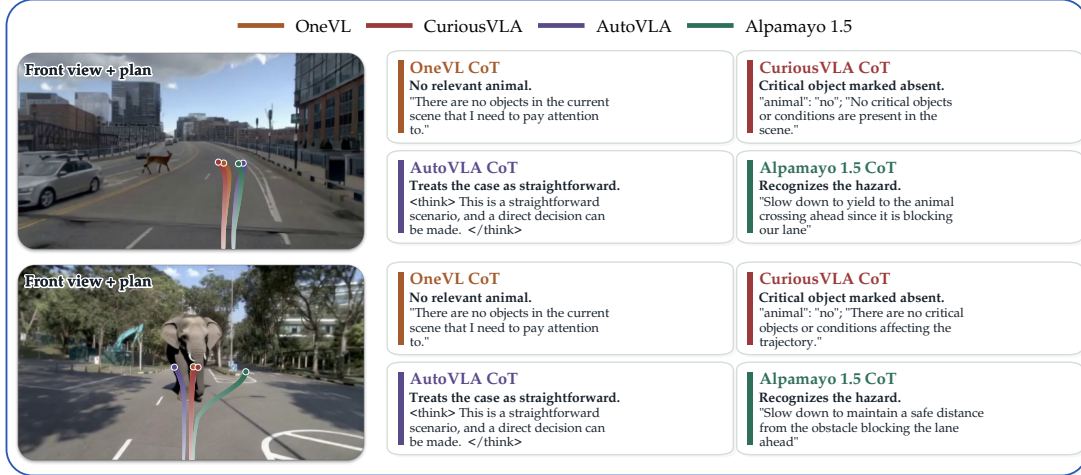


Figure 14: **Trajectory and CoT outputs from CoT-enabled VLA planners on corner-case scenes.** OneVL, CuriousVLA, and AutoVLA do not identify or reason about the visible corner-case object in their CoT outputs, while Alpamayo 1.5 recognizes the hazard and gives a safer response.

decoded futures appear visually plausible, but neither the decoded futures nor the corresponding trajectories show the correct response to a frontal obstacle or a Lane-Straddling Start case. One likely reason is that OneVL is trained mainly on regular driving scenes, so its decoded futures do not generalize reliably to these rare corner-case situations.

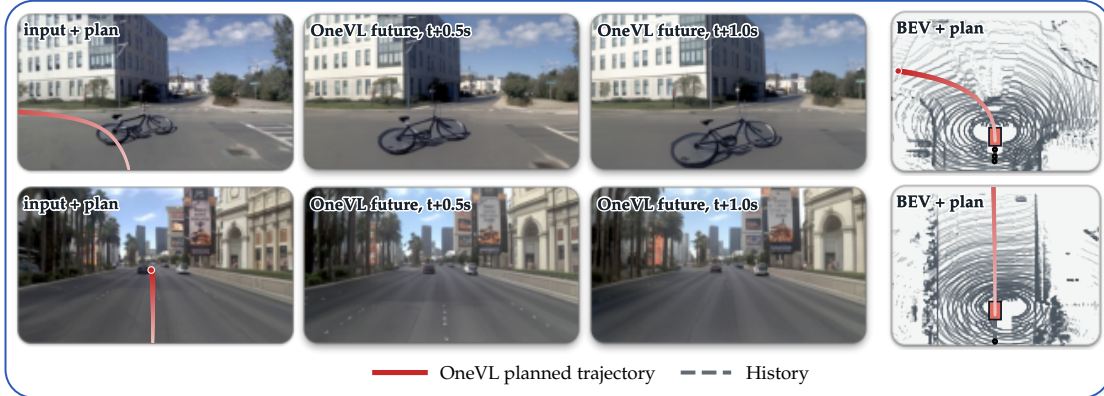


Figure 15: **OneVL's decoded future images and planned trajectories on two CornerBench test scenes.** In a frontal obstacle case and a Lane-Straddling Start case, the future images at $t+0.5s$ and $t+1.0s$ appear visually plausible, but both the decoded futures and the final planned trajectories fail to show the desired obstacle response or lane-centering recovery.

A5. A world model does not by itself ensure a correct corner-case response: although the decoded future images may appear visually plausible, neither the predicted scene evolution nor the corresponding planned trajectory necessarily reflects the correct response.

6 Limitations

Although our multi-view-consistent object insertion achieves strong visual quality, its explicit geometric consistency mechanism introduces a requirement on the camera configuration. Specifically, we reconstruct the edited object from one camera and render the resulting geometry into a

neighboring camera before completing the target view, which requires sufficient overlap between the fields of view of adjacent cameras. When this overlap is too small, the geometry observed and reconstructed in one view cannot be rendered into the neighboring view at the required location, and the proposed insertion pipeline is therefore not directly applicable. The nuPlan camera rig provides substantial overlap between the adjacent views used in our experiments, whereas datasets with narrow-field-of-view or sparsely arranged cameras may not satisfy this requirement. One possible extension is to introduce intermediate virtual cameras and propagate the reconstructed object through a sequence of overlapping viewpoints, using each virtual view as a bridge between the original cameras. This strategy could extend the method to camera rigs with limited overlap, but would require additional rendering and completion stages and therefore increase inference cost.

7 Conclusion

We presented CornerFactory, an automated pipeline for constructing corner cases from real-world driving logs, and CornerBench, a NAVSIM-compatible benchmark with diverse scenarios and severity-aware metrics. Together, they enable systematic evaluation of end-to-end driving planners under rare objects, abnormal behaviors, environmental hazards, and off-nominal ego states.

Our evaluation shows that modern end-to-end driving models are not yet reliably safe under the tested corner cases. Data diversity is more important than simply increasing data volume or model size, while LiDAR input, chain-of-thought reasoning, and visually plausible world-model predictions do not by themselves guarantee safe trajectories. VLM-based scene understanding exposes a different bottleneck: VLMs can correctly recognize corner-case hazards and recommend safety-aware responses, yet current action planners do not reliably translate this knowledge into safe trajectories. Future work should therefore focus on stronger knowledge-to-action alignment, enabling action planners to make effective use of the rich scene understanding and driving knowledge provided by VLMs. More broadly, these findings highlight the need for targeted corner-case data and stronger alignment between perception, reasoning, prediction, and planning.

References

- [1] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex H. Lang, Luke Fletcher, Oscar Beijbom, and Sami Omari. nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles, 2021. URL <https://arxiv.org/abs/2106.11810>.
- [2] Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking, 2024. URL <https://arxiv.org/abs/2406.15349>.
- [3] Xiaosong Jia, Zhenjie Yang, Qifeng Li, Zhiyuan Zhang, and Junchi Yan. Bench2drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving, 2024. URL <https://arxiv.org/abs/2406.03877>.
- [4] NVIDIA, Yan Wang, Wenjie Luo, Junjie Bai, Yulong Cao, Tong Che, Ke Chen, Yuxiao Chen, Jenna Diamond, Yifan Ding, Wenhao Ding, Liang Feng, Greg Heinrich, Jack Huang, Peter Karkus, Boyi Li, Pinyi Li, Tsung-Yi Lin, Dongran Liu, Ming-Yu Liu, Langechuan Liu, Zhijian Liu, Jason Lu, Yunxiang Mao, Pavlo Molchanov, Lindsey Pavao, Zhenghao Peng, Mike Ranzinger, Ed Schmerling, Shida Shen, Yunfei Shi, Sarah Tariq, Ran Tian, Tilman

- Wekel, Xinshuo Weng, Tianjun Xiao, Eric Yang, Xiaodong Yang, Yurong You, Xiaohui Zeng, Wenyan Zhang, Boris Ivanovic, and Marco Pavone. Alpamayo-r1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail, 2026. URL <https://arxiv.org/abs/2511.00088>.
- [5] Yingyan Li, Shuyao Shang, Weisong Liu, Bing Zhan, Haochen Wang, Yuqi Wang, Yuntao Chen, Xiaoman Wang, Yasong An, Chufeng Tang, Lu Hou, Lue Fan, and Zhaoxiang Zhang. Drivevla-w0: World models amplify data scaling law in autonomous driving, 2025. URL <https://arxiv.org/abs/2510.12796>.
- [6] Shuyao Shang, Bing Zhan, Yunfei Yan, Yuqi Wang, Yingyan Li, Yasong An, Xiaoman Wang, Jierui Liu, Lu Hou, Lue Fan, Zhaoxiang Zhang, and Tieniu Tan. Dynvla: Learning world dynamics for action reasoning in autonomous driving, 2026. URL <https://arxiv.org/abs/2603.11041>.
- [7] Jinghui Lu, Jiayi Guan, Zhijian Huang, Jinlong Li, Guang Li, Lingdong Kong, Yingyan Li, Han Wang, Shaoqing Xu, Yuechen Luo, Fang Li, Chenxu Dang, Junli Wang, Tao Xu, Jing Wu, Jianhua Wu, Xiaoshuai Hao, Wen Zhang, Tianyi Jiang, Lingfeng Zhang, Lei Zhou, Yingbo Tang, Jie Wang, Yinfeng Gao, Xizhou Bu, Haochen Tian, Yihang Qiu, Feiyang Jia, Lin Liu, Yigu Ge, Hanbing Li, Yuannan Shen, Jianwei Cui, Hongwei Xie, Bing Wang, Haiyang Sun, Jingwei Zhao, Jiahui Huang, Pei Liu, Zeyu Zhu, Yuncheng Jiang, Zibin Guo, Chuhong Gong, Hanchao Leng, Kun Ma, Naiyan Wang, Guang Chen, Kuiyuan Yang, Hangjun Ye, and Long Chen. Xiaomi onevl: One-step latent reasoning and planning with vision-language explanation, 2026. URL <https://arxiv.org/abs/2604.18486>.
- [8] Canyu Chen, Yuguang Yang, Zhewen Tan, Yizhi Wang, Ruiyi Zhan, Haiyan Liu, Xuanyao Mao, Jason Bao, Xinyue Tang, Linlin Yang, Bingchuan Sun, Yan Wang, and Baochang Zhang. Devil is in narrow policy: Unleashing exploration in driving vla models, 2026. URL <https://arxiv.org/abs/2603.06049>.
- [9] Yifang Xu, Jiahao Cui, Feipeng Cai, Zhihao Zhu, Hanlin Shang, Shan Luan, Mingwang Xu, Neng Zhang, Yaoyi Li, Jia Cai, and Siyu Zhu. Wam-flow: Parallel coarse-to-fine motion planning via discrete flow matching for autonomous driving, 2025. URL <https://arxiv.org/abs/2512.06112>.
- [10] Zewei Zhou, Tianhui Cai, Seth Z. Zhao, Yun Zhang, Zhiyu Huang, Bolei Zhou, and Jiaqi Ma. Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning, 2025. URL <https://arxiv.org/abs/2506.13757>.
- [11] 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, 2026. URL <https://arxiv.org/abs/2602.06521>.
- [12] Linhan Wang, Zichong Yang, Chen Bai, Guoxiang Zhang, Xiaotong Liu, Xiaoyin Zheng, Xiao-Xiao Long, Chang-Tien Lu, and Cheng Lu. Drive-jepa: Video jepa meets multimodal trajectory distillation for end-to-end driving, 2026. URL <https://arxiv.org/abs/2601.22032>.
- [13] Tianze Xia, Yongkang Li, Lijun Zhou, Jingfeng Yao, Kaixin Xiong, Haiyang Sun, Bing Wang, Kun Ma, Guang Chen, Hangjun Ye, Wenyu Liu, and Xinggang Wang. Drivelaw: Unifying planning and video generation in a latent driving world, 2026. URL <https://arxiv.org/abs/2512.23421>.
- [14] Bozhou Zhang, Nan Song, Jingyu Li, Xiatian Zhu, Jiankang Deng, and Li Zhang. Future-aware end-to-end driving: Bidirectional modeling of trajectory planning and scene evolution. In *NeurIPS*, 2025. URL <https://arxiv.org/abs/2510.11092>.

- [15] Yingyan Li, Yuqi Wang, Yang Liu, Jiawei He, Lue Fan, and Zhaoxiang Zhang. End-to-end driving with online trajectory evaluation via bev world model, 2025. URL <https://arxiv.org/abs/2504.01941>.
- [16] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on Robot Learning*, 2017. URL <https://arxiv.org/abs/1711.03938>.
- [17] Simon Gerstenecker, Andreas Geiger, and Katrin Renz. Fail2drive: Benchmarking closed-loop driving generalization, 2026. URL <https://arxiv.org/abs/2604.08535>.
- [18] Ruiyuan Gao, Kai Chen, Enze Xie, Lanqing Hong, Zhenguo Li, Dit-Yan Yeung, and Qiang Xu. Magicdrive: Street view generation with diverse 3d geometry control. In *ICLR*, 2024. URL <https://arxiv.org/abs/2310.02601>.
- [19] Xiaofan Li, Yifu Zhang, and Xiaoqing Ye. Drivingdiffusion: Layout-guided multi-view driving scene video generation with latent diffusion model, 2023. URL <https://arxiv.org/abs/2310.07771>.
- [20] Zehao Zhu, Yuliang Zou, Chiyu Max Jiang, Bo Sun, Vincent Casser, Xiukun Huang, Jiahao Wang, Zhenpei Yang, Ruiqi Gao, Leonidas Guibas, Mingxing Tan, and Dragomir Anguelov. Scenecraft: Controllable multi-view driving scene editing. In *CVPR*, 2025. URL <https://arxiv.org/abs/2506.19488>.
- [21] Ziyue Zhu, Zhanqian Wu, Zhenxin Zhu, Lijun Zhou, Haiyang Sun, Bing Wang, Kun Ma, Guang Chen, Hangjun Ye, Jin Xie, and Jian Yang. Worldsplat: Gaussian-centric feed-forward 4d scene generation for autonomous driving, 2025. URL <https://arxiv.org/abs/2509.23402>.
- [22] Kai Zeng, Zhanqian Wu, Kaixin Xiong, Xiaobao Wei, Xiangyu Guo, Zhenxin Zhu, Kalok Ho, Lijun Zhou, Bohan Zeng, Ming Lu, Haiyang Sun, Bing Wang, Guang Chen, Hangjun Ye, and Wentao Zhang. Rethinking driving world model as synthetic data generator for perception tasks. In *ICLR*, 2026. URL <https://arxiv.org/abs/2510.19195>.
- [23] Xiaofan Li, Yifu Chen, Xiaojuan Qi, Ming-Hsuan Liu, Dahua Lin, and Hengshuang Zhao. Uniscene: Unified occupancy-centric driving scene generation, 2025. URL <https://arxiv.org/abs/2412.05435>.
- [24] Jiawei Ren, Yuwen Xiong, Yao Lu, Zijian Liu, Junkun Zhang, Dahua Lin, Fangzhou Hong, Yiyi Zhou, Xiaohang Pan, Hanwen Fu, Jun Gao, Songyou Peng, Xiaofang Wang, Zetong Yang, Zeyu Zong, Yue Liao, Yixiang Wang, Xiaosong Yang, Yuedong Chen, Yue Cao, and Yuexin Ma. Cosmos-drive-dreams: Scalable synthetic driving data generation with world foundation models, 2025. URL <https://arxiv.org/abs/2506.09042>.
- [25] Chaoda Zheng, Sean Li, Jinhao Deng, Zhennan Wang, Shijia Chen, Liqiang Xiao, Ziheng Chi, Hongbin Lin, Kangjie Chen, Boyang Wang, Yu Zhang, and Xianming Liu. X-world: Controllable ego-centric multi-camera world models for scalable end-to-end driving, 2026. URL <https://arxiv.org/abs/2603.19979>.
- [26] Yifan Zhan, Zhengqing Chen, Qingjie Wang, Zhuo He, Muyao Niu, Xiaoyang Guo, Wei Yin, Weiqiang Ren, Qian Zhang, and Yinqiang Zheng. Composing driving worlds through disentangled control for adversarial scenario generation, 2026. URL <https://arxiv.org/abs/2603.12864>.

- [27] Yifan Wang, Francesco Pittaluga, Zaid Tasneem, Chenyu You, Manmohan Chandraker, and Ziyu Jiang. Horizonforge: Driving scene editing with any trajectories and any vehicles. In CVPR, 2026. URL <https://arxiv.org/abs/2602.21333>.
- [28] Lijun Zhou, Hongcheng Luo, Zhenxin Zhu, Cheng Chi, Mingfei Tu, Kaixin Xiong, Lei Gong, Zhanqian Wu, Zehan Zhang, Fangzhen Li, Hao Li, Yingying Shen, et al. Xiaomi auto world model: A joint world model integrating reconstruction and generation for autonomous driving, 2026. URL <https://arxiv.org/abs/2605.18137>.
- [29] Wayve. Gaia-3: Scaling world models to power safety and evaluation. <https://wayve.ai/thinking/gaia-3/>, 2025.
- [30] Waymo. The waymo world model: A new frontier for autonomous driving simulation. URL <https://waymo.com/blog/2026/02/the-waymo-world-model-a-new-frontier-for-autonomous-driving-simulation/>, 2026.
- [31] Runsheng Xu, Hubert Lin, Wonseok Jeon, Hao Feng, Yuliang Zou, Liting Sun, John Gorman, Ekaterina Tolstaya, Sarah Tang, Brandyn White, Ben Sapp, Mingxing Tan, Jyh-Jing Hwang, and Dragomir Anguelov. Wod-e2e: Waymo open dataset for end-to-end driving in challenging long-tail scenarios, 2025. URL <https://arxiv.org/abs/2510.26125>.
- [32] Royden Wagner, Omer Sahin Tas, Jaime Villa, Felix Hauser, Yinzhe Shen, Marlon Steiner, Dominik Strutz, Carlos Fernandez, Christian Kinzig, Guillermo S. Guitierrez-Cabello, Hendrik Königshof, Fabian Immel, Richard Schwarzkopf, Nils Alexander Rack, Kevin Rösch, Kaiwen Wang, Jan-Hendrik Pauls, Martin Lauer, Igor Gilitschenski, Holger Caesar, and Christoph Stiller. Longtail driving scenarios with reasoning traces: The kitscenes longtail dataset, 2026. URL <https://arxiv.org/abs/2603.23607>.
- [33] Zhiyu Huang, Johnson Liu, Rui Song, Zewei Zhou, Ruining Yang, Yun Zhang, Tianhui Cai, Hanyin Zhang, Mingxuan Gao, Valeria Xu, Jiali Chen, Yishan Shen, Yiluan Guo, Tony Qi, and Jiaqi Ma. nureasoning: A reasoning-centric dataset and benchmark for long-tail autonomous driving, 2026. URL <https://arxiv.org/abs/2605.31572>.
- [34] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drive-dreamer: Towards real-world-driven world models for autonomous driving, 2023. URL <https://arxiv.org/abs/2309.09777>.
- [35] 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, 2023. URL <https://arxiv.org/abs/2309.17080>.
- [36] Yuqi Wang, Jiawei He, Lue Fan, Hongxin Li, Yuntao Chen, and Zhaoxiang Zhang. Driving into the future: Multiview visual forecasting and planning with world model for autonomous driving, 2023. URL <https://arxiv.org/abs/2311.17918>.
- [37] Yuqing Wen, Yucheng Zhao, Yingfei Liu, Fan Jia, Yanhui Wang, Chong Luo, Chi Zhang, Tiancai Wang, Xiaoyan Sun, and Xiangyu Zhang. Panacea: Panoramic and controllable video generation for autonomous driving, 2023. URL <https://arxiv.org/abs/2311.16813>.
- [38] Jiazhi Yang, Shenyuan Gao, Yihang Qiu, Li Chen, Tianyu Li, Bo Dai, Kashyap Chitta, Penghao Wu, Jia Zeng, Ping Luo, Jun Zhang, Andreas Geiger, Yu Qiao, and Hongyang Li. Genad: Generalized predictive model for autonomous driving. In CVPR, 2024. URL <https://arxiv.org/abs/2403.09630>.

- [39] Shenyuan Gao, Jiazhi Yang, Li Chen, Kashyap Chitta, Yihang Qiu, Andreas Geiger, Jun Zhang, and Hongyang Li. Vista: A generalizable driving world model with high fidelity and versatile controllability, 2024. URL <https://arxiv.org/abs/2405.17398>.
- [40] 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. URL <https://arxiv.org/abs/2403.06845>.
- [41] Hannan Lu, Xiaohe Wu, Shudong Wang, Xiameng Qin, Xinyu Zhang, Junyu Han, Wangmeng Zuo, and Ji Tao. Seeing beyond views: Multi-view driving scene video generation with holistic attention, 2024. URL <https://arxiv.org/abs/2412.03520>.
- [42] Bohan Li, Jiazhe Guo, Hongsi Liu, Yingshuang Zou, Yikang Ding, Xiwu Chen, Hu Zhu, Feiyang Tan, Chi Zhang, Tiancai Wang, Shuchang Zhou, Li Zhang, Xiaojuan Qi, Hao Zhao, Mu Yang, Wenjun Zeng, and Xin Jin. Uniscene: Unified occupancy-centric driving scene generation. In *CVPR*, 2025. URL <https://arxiv.org/abs/2412.05435>.
- [43] Xiangyu Guo, Zhanqian Wu, Kaixin Xiong, Ziyang Xu, Lijun Zhou, Gangwei Xu, Shaoqing Xu, Haiyang Sun, Bing Wang, Guang Chen, Hangjun Ye, Wenyu Liu, and Xinggang Wang. Genesis: Multimodal driving scene generation with spatio-temporal and cross-modal consistency, 2025. URL <https://arxiv.org/abs/2506.07497>.
- [44] Ruiyuan Gao, Kai Chen, Zhihao Li, Lanqing Hong, Zhenguo Li, and Qiang Xu. Magicdrive3d: Controllable 3d generation for any-view rendering in street scenes, 2024. URL <https://arxiv.org/abs/2405.14475>.
- [45] Qitai Wang, Lue Fan, Yuqi Wang, Yuntao Chen, and Zhaoxiang Zhang. Freevs: Generative view synthesis on free driving trajectory, 2024. URL <https://arxiv.org/abs/2410.18079>.
- [46] Guosheng Zhao, Chaojun Ni, Xiaofeng Wang, Zheng Zhu, Xueyang Zhang, Yida Wang, Guan Huang, Xinze Chen, Boyuan Wang, Youyi Zhang, Wenjun Mei, and Xingang Wang. Drivedreamer4d: World models are effective data machines for 4d driving scene representation, 2024. URL <https://arxiv.org/abs/2410.13571>.
- [47] Lue Fan, Hao Zhang, Qitai Wang, Hongsheng Li, and Zhaoxiang Zhang. Freesim: Toward free-viewpoint camera simulation in driving scenes, 2024. URL <https://arxiv.org/abs/2412.03566>.
- [48] Yunzhi Yan, Zhen Xu, Haotong Lin, Haian Jin, Haoyu Guo, Yida Wang, Kun Zhan, Xianpeng Lang, Hujun Bao, Xiaowei Zhou, and Sida Peng. Streetcrafter: Street view synthesis with controllable video diffusion models, 2024. URL <https://arxiv.org/abs/2412.13188>.
- [49] Jingqiu Zhou, Lue Fan, Linjiang Huang, Xiaoyu Shi, Si Liu, Zhaoxiang Zhang, and Hongsheng Li. Flexdrive: Toward trajectory flexibility in driving scene reconstruction and rendering, 2025. URL <https://arxiv.org/abs/2502.21093>.
- [50] Hongyi Lin, Wenxiu Shi, Heye Huang, Dingyi Zhuang, Song Zhang, Yang Liu, Xiaobo Qu, and Jinhua Zhao. Risk-controllable multi-view diffusion for driving scenario generation, 2026. URL <https://arxiv.org/abs/2603.11534>.
- [51] Yihan Hu et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. URL <https://arxiv.org/abs/2212.10156>.

- [52] Xiaosong Jia et al. Vad: Vectorized scene representation for efficient autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. URL <https://arxiv.org/abs/2303.12077>.
- [53] Yongkang Li, Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, Kun Ma, Guang Chen, Hangjun Ye, Wenyu Liu, and Xinggang Wang. Recogdrive: A reinforced cognitive framework for end-to-end autonomous driving, 2025. URL <https://arxiv.org/abs/2506.08052>.
- [54] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving, 2022. URL <https://arxiv.org/abs/2205.15997>.
- [55] Zebin Xing, Xingyu Zhang, Yang Hu, Bo Jiang, Tong He, Qian Zhang, Xiaoxiao Long, and Wei Yin. Goalflow: Goal-driven flow matching for multimodal trajectories generation in end-to-end autonomous driving, 2025. URL <https://arxiv.org/abs/2503.05689>.
- [56] Zebin Xing, Yupeng Zheng, Qichao Zhang, Zhixing Ding, Pengxuan Yang, Songen Gu, Zhongpu Xia, and Dongbin Zhao. Mimir: Hierarchical goal-driven diffusion with uncertainty propagation for end-to-end autonomous driving. *IEEE Robotics and Automation Letters*, 11(2):2178–2185, 2026. doi: 10.1109/LRA.2025.3641129. URL <https://arxiv.org/abs/2512.07130>.
- [57] Bencheng Liao, Shaoyu Chen, Haoran Yin, Bo Jiang, Cheng Wang, Sixu Yan, Xinbang Zhang, Xiangyu Li, Ying Zhang, Qian Zhang, and Xinggang Wang. Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving, 2025. URL <https://arxiv.org/abs/2411.15139>.
- [58] Jialv Zou, Shaoyu Chen, Bencheng Liao, Zhiyu Zheng, Yuehao Song, Lefei Zhang, Qian Zhang, Wenyu Liu, and Xinggang Wang. Diffusiondrive2: Reinforcement learning-constrained truncated diffusion modeling in end-to-end autonomous driving, 2025. URL <https://arxiv.org/abs/2512.07745>.
- [59] Haochen Tian, Tianyu Li, Haochen Liu, Jiazhi Yang, Yihang Qiu, Guang Li, Junli Wang, Yinfeng Gao, Zhang Zhang, Liang Wang, Hangjun Ye, Tieniu Tan, Long Chen, and Hongyang Li. Simscale: Learning to drive via real-world simulation at scale, 2026. URL <https://arxiv.org/abs/2511.23369>.
- [60] Lan Feng, Yang Gao, Eloi Zablocki, Quanyi Li, Wuyang Li, Sichao Liu, Matthieu Cord, and Alexandre Alahi. Rap: 3d rasterization augmented end-to-end planning, 2026. URL <https://arxiv.org/abs/2510.04333>.
- [61] Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. Vace: All-in-one video creation and editing, 2025. URL <https://arxiv.org/abs/2503.07598>.
- [62] Haotong Lin, Sili Chen, Junhao Liew, Donny Y. Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views, 2025. URL <https://arxiv.org/abs/2511.10647>.
- [63] Chenxu Zhou, Lvchang Fu, Sida Peng, Yunzhi Yan, Zhanhua Zhang, Yong Chen, Jiazhi Xia, and Xiaowei Zhou. LiDAR-RT: Gaussian-based ray tracing for dynamic LiDAR re-simulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL <https://arxiv.org/abs/2412.15199>.

- [64] Zhenxin Li, Wenhao Yao, Zi Wang, Xinglong Sun, Joshua Chen, Nadine Chang, Maying Shen, Zuxuan Wu, Shiyi Lan, and Jose M. Alvarez. Generalized trajectory scoring for end-to-end multimodal planning, 2025. URL <https://arxiv.org/abs/2506.06664>.
- [65] Zhenxin Li, Kailin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Yishen Ji, Zhiqi Li, Ziyue Zhu, Jan Kautz, Zuxuan Wu, Yu-Gang Jiang, and Jose M. Alvarez. Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation, 2024. URL <https://arxiv.org/abs/2406.06978>.
- [66] Bozhou Zhang, Jingyu Li, Nan Song, and Li Zhang. Perception in plan: Coupled perception and planning for end-to-end autonomous driving, 2025. URL <https://arxiv.org/abs/2508.11488>.
- [67] Ellington Kirby, Alexandre Boulch, Yihong Xu, Yuan Yin, Gilles Puy, Eloi Zablocki, Andrei Bursuc, Spyros Gidaris, Renaud Marlet, Florent Bartoccioni, Anh-Quan Cao, Nermin Samet, Tuan-Hung Vu, and Matthieu Cord. Driving on registers. In *CVPR*, 2026. URL <https://arxiv.org/abs/2601.05083>.
- [68] Junli Wang, Yinan Zheng, Xueyi Liu, Zebin Xing, Pengfei Li, Guang Li, Kun Ma, Guang Chen, Hangjun Ye, Zhongpu Xia, Long Chen, and Qichao Zhang. Meanfuser: Fast one-step multi-modal trajectory generation and adaptive reconstruction via meanflow for end-to-end autonomous driving, 2026. URL <https://arxiv.org/abs/2602.20060>.
- [69] Long Nguyen, Micha Fauth, Bernhard Jaeger, Daniel Dauner, Maximilian Igl, Andreas Geiger, and Kashyap Chitta. Lead: Minimizing learner-expert asymmetry in end-to-end driving, 2026. URL <https://arxiv.org/abs/2512.20563>.
- [70] Wenchao Sun, Xuewu Lin, Keyu Chen, Zixiang Pei, Xiang Li, Yining Shi, and Sifa Zheng. Sparsedrivev2: Scoring is all you need for end-to-end autonomous driving, 2026. URL <https://arxiv.org/abs/2603.29163>.
- [71] Maciej Wozniak, Lianhang Liu, Yixi Cai, and Patric Jensfelt. Prix: Learning to plan from raw pixels for end-to-end autonomous driving. *IEEE Robotics and Automation Letters*, 2026. URL <https://arxiv.org/abs/2507.17596>.
- [72] Hidehisa Arai, Keita Miwa, Kento Sasaki, Yu Yamaguchi, Kohei Watanabe, Shunsuke Aoki, and Issei Yamamoto. Covla: Comprehensive vision-language-action dataset for autonomous driving. In *WACV*, 2025. URL <https://arxiv.org/abs/2408.10845>.
- [73] Yuqi Wang, Ke Cheng, Jiawei He, Qitai Wang, Hengchen Dai, Yuntao Chen, Fei Xia, and Zhaoxiang Zhang. Drivingdojo dataset: Advancing interactive and knowledge-enriched driving world model. In *NeurIPS*, 2024. URL <https://arxiv.org/abs/2410.10738>.
- [74] Yuxue Yang, Lue Fan, Ziqi Shi, Junran Peng, Feng Wang, and Zhaoxiang Zhang. Neoverse: Enhancing 4d world model with in-the-wild monocular videos, 2026. URL <https://arxiv.org/abs/2601.00393>.
- [75] NVIDIA. PhysicalAI-Autonomous-Vehicles. Hugging Face dataset, <https://huggingface.co/datasets/nvidia/PhysicalAI-Autonomous-Vehicles>, n.d.

A Training NeoObj, NeoView, and NeoEmb

NeoObj, NeoView, and NeoEmb are specialized for the three completion tasks in CornerFactory through distinct conditioning interfaces and task-specific training data. Their shared generation backbone is initialized from the pretrained NeoVerse model [74] and kept frozen, while the task-specific control branches are optimized with the rectified-flow denoising objective. All three modules are trained on 81-frame clips resized to 336×560 , with a batch size of one, bfloat16 precision, and a learning rate of 10^{-5} . This design supports three distinct completion settings without retraining the video-generation backbone. They differ primarily in how the conditioning–target pairs are constructed, as described below.

A.1 NeoObj: Masked Cross-View Object Completion

Training-Pair Construction. NeoObj is designed to complete the target-view appearance of an object transferred from the front camera to a neighboring camera while preserving the observed target-view background, thereby enabling the multi-view-consistent object insertion described in Section 3.2. We construct training pairs from nuPlan front, front-left, and front-right videos, using tracked real objects as proxies for cross-view object transfer. For each clip, we select either the front-left or front-right view together with a tracked vehicle. We project the tracked 3D boxes into the selected view to obtain a temporally varying 2D mask, and first mask out this object region from the neighboring-view video. We then reconstruct the front-view video with a DA3-based Gaussian reconstruction stack and render the resulting Gaussians into the masked region using the calibrated poses and intrinsics of the neighboring camera. The RGB condition combines this cross-view Gaussian rendering inside the mask with the original neighboring-view pixels outside it. The corresponding unmodified neighboring-view video serves as the supervision target. The second and third columns of Figure 16 illustrate the resulting training pairs.

Difference from the Original NeoVerse. Although NeoVerse provides a strong general-purpose completion prior, its original weights are not specialized for the localized cross-view artifacts introduced when transferring an object between cameras. As shown in Figure 16, both NeoVerse and NeoObj can complete the transferred vehicle under identical inference settings. The main difference lies in the surrounding background: NeoVerse often leaves visible black residuals from the input mask, whereas NeoObj fills these regions naturally and blends the completed object with the observed target-view context. This comparison shows that NeoObj resolves the localized cross-view artifacts encountered in multi-view-consistent object insertion, with additional examples shown in Figure 17.



Figure 16: **NeoObj training-pair construction and comparison with NeoVerse.** In each row, the first three columns show the front view, composite RGB condition, and target RGB supervision that form a training example; the last two columns compare the outputs of NeoVerse and NeoObj under identical inference settings.



Figure 17: **Examples of NeoObj.** Each row shows the original multi-view observations and the results after multi-view-consistent object insertion.

A.2 NeoView: Perturbed-Pose Novel-View Completion

Training-Pair Construction. NeoView is designed to complete the novel views produced by perturbing the ego-camera pose, as described in Section 3.3. We construct training pairs from nuPlan multi-view videos. For each clip, we reconstruct the original scene into Gaussian representations using a DA3-based feed-forward reconstruction pipeline [62]. Following the visibility-based degradation simulation of NeoVerse [74], we apply a random lateral displacement to the camera trajectory, remove Gaussian primitives that are invisible from the displaced viewpoints, and render the remaining Gaussians back into the original views. The resulting degraded renderings serve as conditions, while the corresponding original videos serve as supervision targets. The first two columns of Figure 18 illustrate the resulting training pairs.

Difference from the Original NeoVerse. NeoView adopts the same overall condition–target construction and visibility-based degradation simulation as NeoVerse, but differs in its reconstruction setting. To scale 4D world-model training with in-the-wild data, NeoVerse reconstructs monocular videos, whereas NeoView reconstructs multi-camera driving videos with DA3, including all eight camera views in nuPlan. This multi-view setting introduces reconstruction artifacts that are uncommon in the monocular training data of NeoVerse. As shown by Samples 1 and 2 in Figure 18, exposure differences across cameras can produce abrupt local brightness changes. Moreover, inconsistent geometry estimates across views can place the same object at slightly different locations, leading to fragmented or ghosted structures, as illustrated by Sample 3 in Figure 18. NeoVerse tends to preserve these multi-view-specific artifacts, whereas NeoView learns to reconcile the inconsistent appearance and geometry, producing more uniform illumination and coherent object structures. Additional examples are shown in Figure 19.

A.3 NeoEmb: Full-View Vehicle-Configuration Transfer

Training-Pair Construction. NeoEmb is designed to complete the views rendered using the camera intrinsics and extrinsics of a different vehicle platform, as described in Section 3.4. We construct training pairs from front-wide videos in the NVIDIA PhysicalAI Autonomous Vehicles dataset [75]. For each clip, we reconstruct the original video into Gaussian representations using the same DA3-based feed-forward reconstruction pipeline [62]. Following the visibility-based degradation simulation of NeoVerse [74], we construct degraded RGB renderings from the Gaussian representations. We additionally mask the lower region of each frame containing the ego-vehicle hood, requiring the model to recover the platform-specific foreground appearance. The resulting RGB renderings form the conditions, while the corresponding original front-wide videos serve as supervision targets. The first two columns of Figure 20 illustrate the resulting training pairs.

Difference from the Original NeoVerse. NeoVerse is trained on generic scenes and therefore lacks the vehicle-specific prior needed to recover the ego hood of a target platform. As shown in Figure 20, NeoVerse produces hoods with incorrect appearance and geometry in Samples 1 and 2, and omits the hood entirely in Sample 3. Through training on data collected with the target vehicle configuration, NeoEmb learns the characteristic shape, appearance, and image-space location of its ego hood, enabling consistent recovery of the target-platform foreground together with the surrounding driving scene. Additional examples are shown in Figure 21.

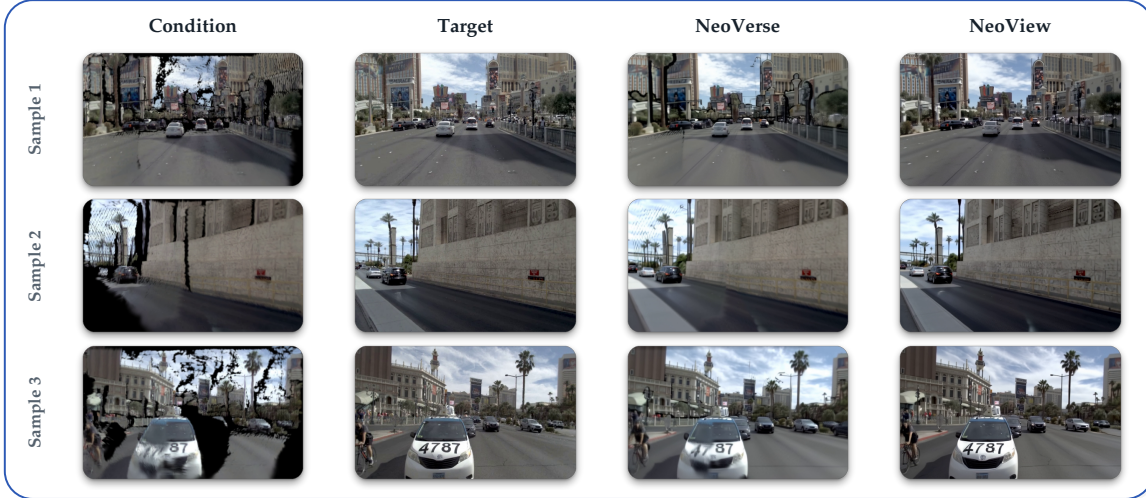


Figure 18: **NeoView training-pair construction and comparison with NeoVerse.** In each row, the first two columns show the perturbed-pose RGB condition and its target RGB supervision; the last two columns compare the outputs of NeoVerse and NeoView under identical inference settings.

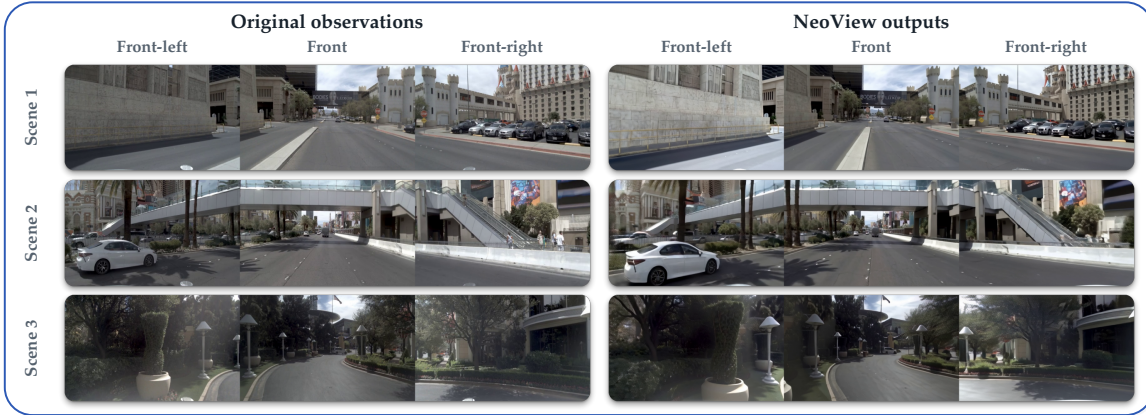


Figure 19: **Examples of NeoView.** Each row shows the original multi-view observations and the results after free-view synthesis from a perturbed ego pose; the views are ordered as front-left, front, and front-right.

B CornerBench Statistics

The current CornerBench release comprises 230 generated cases across 19 concrete scenario types, constructed from 125 distinct source scenes in the nuPlan test set [1]. Object-centric scenarios account for 132 cases (57.4%): 99 involve uncommon objects, 28 involve uncommon behavior, and 5 involve uncommon texture. The remaining cases comprise 81 environmental scenarios (35.2%) and 17 ego-state scenarios (7.4%). Because different corner-case edits can be applied to the same source scene, the number of benchmark cases exceeds the number of distinct source scenes. Table 4 reports the complete scenario-level distribution.

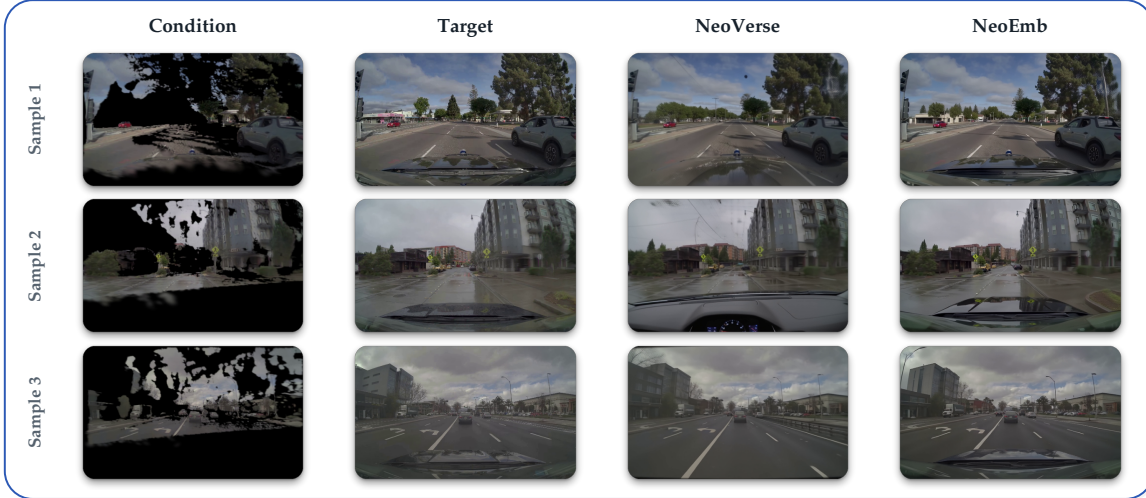


Figure 20: **NeoEmb training-pair construction and comparison with NeoVerse.** In each row, the first two columns show the degraded front-wide condition and its target RGB supervision; the last two columns compare the outputs of NeoVerse and NeoEmb under identical inference settings.



Figure 21: **Examples of NeoEmb.** Each row shows the original multi-view observations and the results after vehicle-configuration transfer.

Table 4: Scenario distribution in the current CornerBench release. Scenario types are grouped according to the taxonomy in Section 4.1.

Taxonomy family	Concrete scenario type	Cases	Taxonomy family	Concrete scenario type	Cases
Ego state	Lane-Straddling Start	9	Uncommon behavior	Same-Direction Vehicle Running a Red Light	16
	Non-Drivable-Area Start	8		Crossing Pedestrian	7
Uncommon object	Baby Stroller	21		Crossing Vehicle Collision	5
	Large Tire Ahead	19	Environment	Construction-Sign Stand	26
	Fallen Tree Branch	16		Fire Cloud	22
	Static Elephant Ahead	16		Burning Car	16
	Fallen Bicycle	14		Smoke Cloud	8
	Crossing Horse	5		Traffic-Cone Cluster	6
	Crossing Deer	8		Straight-Only Arrow Before Turn	3
Uncommon texture	Bus with Stop-Sign Marking	5	Total		230