NeuroNCAP: Photorealistic Closed-Loop Safety Testing for Autonomous Driving

William Ljungbergh[1](✉), Adam Tonderski[1], Joakim Johnander[1], Holger Caesar[4], Kalle Åström[3], Michael Felsberg[2], and Christoffer Petersson[1]

[1] Zenseact, Gothenburg, Sweden
`william.ljungbergh@liu.se`
[2] Linköping Univeristy, Linköping, Sweden
[3] Lund University, Lund, Sweden
[4] Delft University of Technology, Delft, Netherlands
`https://github.com/atonderski/neuro-ncap`

Abstract. We present a versatile NeRF-based simulator for testing autonomous driving (AD) software systems, designed with a focus on sensor-realistic closed-loop evaluation and the creation of safety-critical scenarios. The simulator learns from sequences of real-world driving sensor data and enables reconfigurations and renderings of new, unseen scenarios. In this work, we use our simulator to test the responses of AD models to safety-critical scenarios inspired by the European New Car Assessment Programme (Euro NCAP). Our evaluation reveals that, while state-of-the-art end-to-end planners excel in nominal driving scenarios in an open-loop setting, they exhibit critical flaws when navigating our safety-critical scenarios in a closed-loop setting. This highlights the need for advancements in the safety and real-world usability of end-to-end planners. By publicly releasing our simulator and scenarios as an easy-to-run evaluation suite, we invite the research community to explore, refine, and validate their AD models in controlled, yet highly configurable and challenging sensor-realistic environments.

Keywords: Autonomous driving · Closed-loop simulation · Trajectory planning · Neural rendering

1 Introduction

Recent work on autonomous driving (AD) [22,23] suggests designing and training a holistic neural network for mapping sensor inputs directly to a planned trajectory. Compared to prior work that used modular software stacks, engineered interfaces between modules, or handcrafted rules, this end-to-end approach has several advantages. First, as the driving behavior is learned, the predicted trajectories are expected to resemble how a typical human driver would act. Second,

W. Ljungbergh and A. Tonderski—Equal contribution.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-73404-5_10.

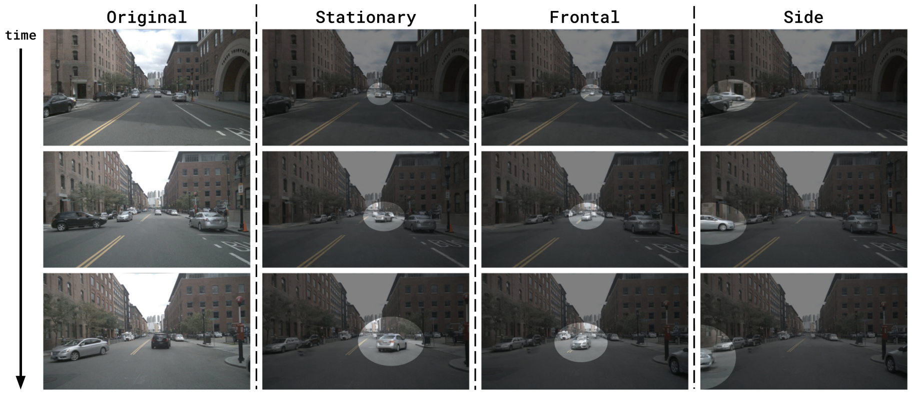


Fig. 1. The core idea in NeuroNCAP is to leverage NeRFs to realistically simulate many safety-critical scenarios from a sequence of real-world data. Here we show the original scenario, followed by examples of our three types of collision scenarios: stationary, frontal, and side. The inserted safety-critical actor has been highlighted for illustration purposes. We can generate hundreds of unique scenarios from each log by selecting different actors, jittering their trajectories, and choosing different starting conditions for the ego vehicle. Note that scenarios are not pre-generated, but rather obtained by iteratively generating new images, computing a plan, and acting upon said plan.

the approach is scalable in the sense that more data leads to more robust as well as generalizable driving performance [4, 23] and in the sense that there is no need to manually design intermediate interfaces or cost functions. The neural network may be divided into modules, but the interfaces between them are learned in order to mitigate information loss (Fig. 1).

Hu *et al.* [22] demonstrated that their end-to-end planner, UniAD, performed well on the popular nuScenes [5] planning benchmark. This is an open-loop benchmark, where the tested planner never influences the driving. Instead, the plans are compared to the trajectory taken by the vehicle during data collection and a score is computed based on the similarity between the two. Codevilla *et al.* [8], as well as Dauner *et al.* [11], shed some doubt about the correlation between such an open-loop score and the actual driving performance. This begs the question, how would state-of-the-art end-to-end planners fare if their predicted policy would be acted upon? Unlike regular planners that can be evaluated in a closed-loop manner using straightforward object-level simulations, end-to-end planners require complex sensor simulations to accurately predict their behavior in real-world scenarios. This introduces significant challenges due to the complexity and computational demands of high-fidelity sensor simulation. Moreover, the nuScenes benchmark contains normal driving scenarios, in which no collisions occur. It is unclear how state-of-the-art end-to-end planners would perform in safety-critical scenarios, where a crash is likely unless swift corrective action is undertaken.

In this work, we subject state-of-the-art end-to-end planners to closed-loop evaluation in safety-critical scenarios. Given sensor data, planners predict a plan.

The plan is then executed under the constraints of a vehicle model in order to propagate the state of the ego-vehicle forward in time. Given the new state, we use recent advances in neural rendering—NeRFs—to resolve the problem of generating realistic sensor data. These three steps are then repeated until either a crash occurs or we deem the scenario to be over. By executing the predicted plan, we aim to reduce the gap between model evaluation and deployment. It is possible that the neural renderer exacerbates the gap, if the rendered images are not sufficiently realistic. We quantify to what extent the renderer affects the gap by analyzing the intermediary perception outputs that are available in state-of-the-art (SotA) end-to-end planners, such as UniAD [22] and VAD [23]. Moreover, we ensure that replacing the real sensor data with rendered sensor data during open-loop evaluation does not negatively affect the open-loop performance.

To generate safety-critical scenarios, we take inspiration from the European New Car Assessment Protocol (Euro NCAP) for collision avoidance [14]. This protocol comprises several scenario types that have been identified as safety-critical. These scenario types are rare, but are likely to lead to a collision unless the planner properly deals with them. We craft a subset of these scenarios by altering recordings of scenes from the nuScenes dataset [5]. We evaluate the driving quality by whether there is a crash, and at what velocity that crash occurs. Our benchmark should be viewed as a necessary but not sufficient condition for high quality driving. To summarize, our contributions are as follows:

1. We release an open source framework for photorealistic closed-loop simulation for autonomous driving.
2. We construct safety-critical scenarios, inspired by the industry standard Euro NCAP, that cannot safely be collected in the real world.
3. Using the simulator and our scenarios, we design a novel evaluation protocol that focuses on collisions rather than displacement metrics.
4. We show that two SotA end-to-end planners fail severely in our safety-critical scenarios despite accurately perceiving the environment.

2 Related Work

End-to-End Driving Models: The autonomous driving task has traditionally been divided into different modules—e.g., perception, prediction, and planning—that are constructed individually [22,23,30]. Hu *et al.* [22] argue that this division comes with a number of disadvantages: information loss across modules, error accumulation, and feature misalignment. Jiang *et al.* [23] emphasize that a planning module might need access to the semantic information of the sensor data that would not be present in a hand-crafted interface. These two works proceed to argue in favor of end-to-end planning. The pioneering work of Pomerlau *et al.* [32] proposes such a planner, where a single neural network is trained to map sensor input to an output trajectory. Decades of neural network advancements sparked new interest in end-to-end planning [4,7,9,10,33]. The black-box nature of these planners, however, makes them difficult to optimize and their results hard to interpret [23]. Hu *et al.* [22] and Jiang *et al.* [23] propose two end-to-end neural network planners with intermediary outputs, corresponding to those of a modular approach. Their planners are divided into modules, but the module interfaces are learned, consisting of deep feature vectors.

Open-Loop Evaluation of End-to-End Planners: Pomerleau *et al.* [32] evaluated their driving model by letting it drive a real-world test-vehicle. Such a setup makes large-scale testing costly and results may be hard to reproduce. Recent works on end-to-end planning [22,23] instead evaluate in *open loop*, where models predict a plan based on recorded sensor data. The predicted plans are never executed and instead, the actions are fixed to whatever was recorded. This setup has also been used in works on object-level planning [12,15,27], which assume perfect perception and feed both a map of the static environment as well as tracks of dynamic objects into the model. Such open-loop evaluation constitutes a gap between evaluation and real-world deployment. Moreover, performance is usually measured as the distance between the predicted plan and the trajectory driven by the vehicle in the recording [12,15,22,23,27]. While an error of zero corresponds to human-level driving, it is not necessarily true that a lower error is better. This can be realized by considering a scenario where two different trajectories are equally good. These issues were studied by Codevilla *et al.* [8] and they found that open-loop evaluation is not necessarily correlated with actual driving quality. Dauner *et al.* [11] draw similar conclusions.

Closed-Loop Evaluation and Simulation: Given the aforementioned issues of open-loop evaluation, closed-loop simulation becomes attractive. Several object-level simulators have been proposed [2,6,17,25]. These simulators do not generate sensor data, however, which makes it impossible to test end-to-end planners in closed loop. A number of hand-crafted graphical simulators have been proposed [13,35,36]. The challenge for such simulators is twofold: it is difficult to create realistic-looking images and it is hard to create graphical assets that capture the variety of the real world. Work on world models [18,39] demonstrate that the future of a scene—e.g., an Atari game—can be predicted in latent space and that vectors in latent space can be decoded into sensor input. Hu *et al.* [20] build a world-model from a large-scale real-world automotive dataset. Amini *et al.* [3] propose VISTA, in which novel views can be synthesized around the local trajectory by unprojecting the closest image via predicted depth, and reprojecting. Yang *et al.* [42] propose to use neural radiance fields (NeRF) to create photorealistic sensor input of a scene. The method was subsequently improved by Tonderski *et al.* [37] with more accurate sensor modeling and higher rendering quality, particularly for the 360-degree setting considered here.

New Car Assessment Programs: The New Car Assessment Program (NCAP) was introduced in 1979 by the U.S. Department of Transportation's National Highway Safety Administration in order to provide consumers with information on the relative safety potential of automobiles [19]. NCAP crash-tested vehicles and scored vehicles based on the probability of serious injury. A similar European protocol was proposed in 1996, the European New Car

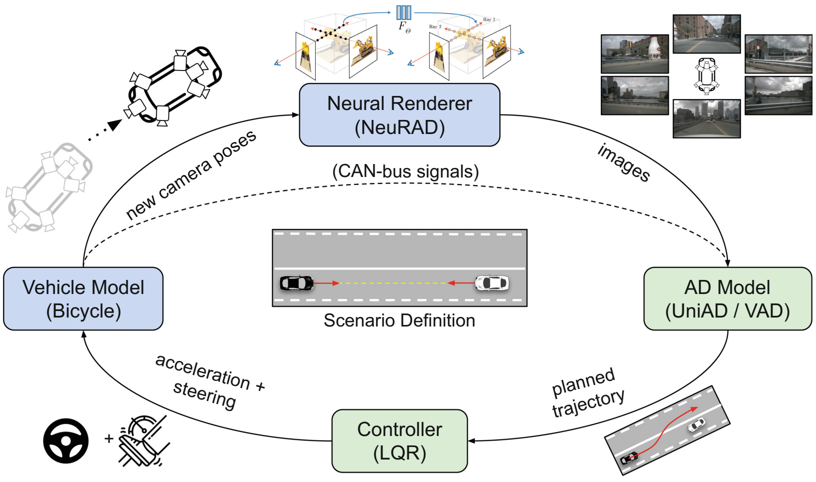


Fig. 2. Our closed-loop simulation engine comprises four parts. First, given a driving log, a neural renderer (NeRF) provides photo-realistic images given the ego-vehicle state. Second, an AD model (e.g., the end-to-end planner UniAD [22]) uses these to predict a future ego-trajectory. Third, a controller estimates acceleration and steering signals. Finally, a vehicle model propagates the ego-vehicle state one step into the future. This process is then iterated to achieve closed-loop simulation. Blue indicates simulator, green indicates AD system. (Color figure online)

Assessment Programme (Euro NCAP). In 2009, Euro NCAP was overhauled in order to also include tests of emerging crash avoidance systems [38]. Initially, this included electronic stability control and speed assistance systems, but this was later extended to include additional systems, such as autonomous emergency braking [38] and autonomous emergency steering [14]. In this work, we take inspiration from the Euro NCAP automatic collision avoidance assessment protocol [14]. This protocol provides scenarios in which a crash will occur unless action is taken. To obtain a full score, the vehicle needs to brake or steer to avoid the accident. Partial scores are awarded if the impact velocity is sufficiently reduced.

3 Method

Our end-to-end planning evaluation protocol comprises a closed-loop-simulator (see Sect. 3.1) and a collision-focused evaluation protocol (see Sect. 3.2).

3.1 Closed-Loop Simulator

Our closed-loop simulator repeatedly performs four steps. First, high-quality camera input is rendered given the state and camera calibration of the ego vehicle. The renderer is constructed from a log of a driving vehicle. Second, the end-to-end planner predicts a future ego-vehicle trajectory given the rendered camera input and the ego-vehicle state. Third, a controller converts the planned trajectory to a set of control inputs. Fourth, a vehicle model propagates the ego-state forward in time given the control inputs. This procedure is illustrated in Fig. 2. Next, we elaborate on each of the four steps.

Neural Renderer: In order to simulate novel sensor data, we adopt a neural renderer [29]. NeRFs learn an implicit representation of the 3D environment from a log of collected real-world data. Once trained, NeRFs can render sensor-realistic novel views from said scene. Recent advances add the ability to also edit the dynamic objects in the scene by changing their corresponding 3D bounding boxes [31]. Specifically, actors can be removed, added, or set to follow novel trajectories, which in our case enables the creation of safety-critical scenarios. For example, to simulate a rare but critical safety scenario, a vehicle that is originally moving in an adjacent lane can be positioned to be stationary and in the same lane as the ego-vehicle. This novel situation necessitates the ego-vehicle to either brake or execute a precise overtaking maneuver.

Two things should be noted. First, the recently proposed NeuRAD [37] also supports the rendering of LiDAR data. However, as state-of-the-art end-to-end planners consume only camera data, we focus only on camera data in this work. Second, as we show in our experiments, the domain gap introduced by modern NeRFs compared to real data is sufficiently small for the perception parts of end-to-end planners to still function with high performance. However, we expect this gap to be reduced further with future developments in neural rendering.

AD Model: Recent works on end-to-end planning [21–23] describe a system that consumes (i) raw sensor data; (ii) the ego-vehicle state; and (iii) a high-level plan to predict a planned trajectory. The planned trajectory comprises waypoints at some frequency and with some time horizon. It should be noted that while we primarily aim to analyze state-of-the-art end-to-end planners, this module can be replaced with any type of planner, e.g., a modular detector-tracker-planner pipeline.

Controller: In order to apply the vehicle model, the waypoints need to be converted to a sequence of control signals, corresponding to a sequence of steering angle (δ) and acceleration (a) commands. Following Caesar *et al.* [24], we achieve this with a linear quadratic regulator (LQR). Note that while we only analyze planners that output waypoints, the planner could instead directly output a sequence of control signals.

Vehicle Model: Given a set of control signals, generated from the planned trajectory, the vehicle state is propagated through time. To this end, we follow prior closed-loop simulators [6, 24] and adopt a discrete version of the kinematic bicycle model [34]. It can formally be described as

$$S = \begin{pmatrix} x \\ y \\ \theta \\ v \end{pmatrix}, \quad \frac{dS}{dt} = \begin{pmatrix} v\cos\theta \\ v\sin\theta \\ \frac{v\tan\delta}{L} \\ a \end{pmatrix} . \tag{1}$$

The state S is composed of x, y, θ, and v, where x and y is the longitudinal and lateral position; θ the rotation; and v the speed of the vehicle. Furthermore, L is the wheelbase of the vehicle, and δ and a are the control signals. We adopt control signal limits and the wheelbase based on the vehicle that was used when collecting the data. Note that x, y, and θ live in a global coordinate system, whereas v, L, δ, and a are values. Further, we limit acceleration to ± 6 m/s^2, matching maximum braking in EuroNCAP CCRb [14] and in line with [26].

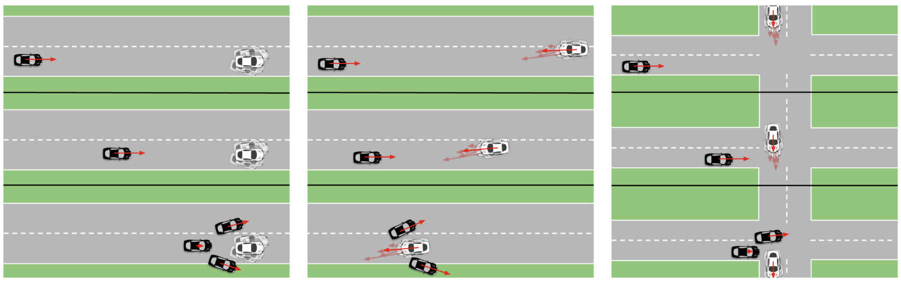
3.2 Evaluation

In contrast to common evaluation practices—*i.e.*, averaging performance across large-scale datasets—we instead focus our evaluation on a small set of carefully designed safety-critical scenarios. These scenarios have been crafted such that any model that cannot successfully handle all of them, should be considered unsafe. We have taken inspiration from the industry standard Euro NCAP testing [14] (see Sect. 2) and define three types of scenarios, each characterized by the behavior of the actor that we are about to collide with: *stationary*, *frontal*, and *side*. Following the Euro NCAP nomenclature, we refer to this actor as the *target actor*. The aim is to control the ego-vehicle to avoid a collision with the target actor or at least decrease the collision velocity. A collision event is defined as having nonzero overlap between the bounding boxes of the ego vehicle and the target actor.

For each scenario type, we create multiple scenarios. Each scenario is based on data collected from around 20 s of real-world driving. The ego-vehicle and target actor states are initialized such that if current speeds and steering angles are maintained, a collision will occur at approximately 4 s into the future. All non-stationary actors are removed from the scene and we randomly select one of these to be the target actor, taking into consideration whether the actor has been observed sufficiently closely, and under the necessary angles, to produce realistic renderings. As our renderer is limited to rigid actors, we exclude pedestrians from this selection. Finally, we randomly jitter the position, rotation, and velocity of the target actor within scenario-specific intervals. During evaluation, we run each scenario for a large number of runs (with a fixed random seed) and compute average results. Next, we describe the characteristics of each type of scenario.

Stationary: This is a relatively simple type of scenario where a stationary target actor has been placed in the ego-lane. The target vehicle can be placed with an arbitrary rotation, but will remain stationary throughout the scenario. This means that the ego-vehicle can either commit to a harsh break or a steering maneuver to avoid a crash. See Fig. 3a for an illustration.

Frontal: The frontal scenarios comprise a target actor that is driving in the opposite direction and has drifted over into the ego-lane on a collision path with the ego-vehicle. Thus, the ego-vehicle cannot avoid crashing by breaking, only reducing the impact speed. To completely avoid a collision, the ego-vehicle must instead perform a steering maneuver. See Fig. 3b for an illustration.



(a) Stationary scenario. (b) Frontal scenario. (c) Side scenario.

Fig. 3. Illustration of the different scenario types used in the NeuroNCAP evaluation protocol. There are multiple actions that can lead to a successfully completed scenario e.g., harsh breaking or a steering maneuver. To increase the robustness of the test and allow for multiple runs, we introduce small random perturbations to the target actor.

Side: The side-collision scenarios feature a target actor crossing our lane from a perpendicular direction. If the current velocity of the ego-vehicle is maintained, there will be a side-collision. The ego-vehicle can avoid collision either by braking for the oncoming target actor, or by conducting a slight steering maneuver while speeding past the target actor. See Fig. 3c for an illustration.

NeuroNCAP Score: For each scenario, a score is computed. A full score is achieved only by completely avoiding collision. Partial scores are awarded by successfully reducing the impact velocity. In spirit of the 5-star Euro NCAP rating system [14] we compute the NeuroNCAP score (NNS) as

$$\text{NNS} = \begin{cases} 5.0 & \text{if no collision} \\ 4.0 \cdot \max(0, 1 - v_i/v_r) & \text{otherwise} \end{cases}, \tag{2}$$

where v_i is the impact speed as the magnitude of relative velocity between ego-vehicle and colliding actor, and v_r is the reference impact speed that would occur if no action is performed. In other words, the score corresponds to a 5-star rating if collision is entirely avoided, and otherwise the rating is linearly decreased from four to zero stars at (or exceeding) the reference impact speed.

4 Experiments

First, we start by outlining the details of our experiments in Sect. 4.1. Next, we show the quantitative results from our NeuroNCAP evaluation in Sect. 4.2 and some qualitative examples in Sect. 4.3. Last, we show results from our real-to-sim study in Sect. 4.4, building more confidence in the results from the NeuroNCAP evaluation.

4.1 Experimental Setting

Dataset: While there are many datasets targeting autonomous driving [1,16, 28,40,41], nuScenes [5] has received the most widespread adaptation for end-to-end planning. It features urban environments with highly interactive scenarios, making it suitable for our safety-critical scenario generation. Thanks to its widespread adaptation it also allows us to use official implementations and network weights of the models we evaluate. NuScenes is divided into 1000 sequences, out of which 150 are reserved for validation. From these 150, we choose 14 diverse sequences—deemed to be suitable based on the behavior of agents present in the scene—to serve as the basis for our safety-critical scenarios.

Scenarios: Each scenario is designed by hand, considering which actors are suitable for the given sequence, the most reasonable collision trajectories, as well as defining allowed ranges for the different kinds of randomization. During evaluation we run each scenario 100 times (with fixed random seed) and average the results. Not all sequences can be used for all types of scenarios, as for instance we cannot simulate a realistic side collision on a single straight road. We therefore select suitable sequences for each scenario type. For more details, and qualitative examples of each scenario, we refer to the supplementary material.

Neural Renderer: As our renderer, we opt to use NeuRAD [37], a SotA neural renderer developed specifically for autonomous driving and verified to work well with nuScenes. As we wish to maximize the reconstruction quality, we use the larger configuration (NeuRAD-L), and train for 100k steps with default hyper-parameters. As pose information in nuScenes is limited to the bird's eye view plane, we employ pose optimization to recover the missing information. Finally, we adopt actor flipping along the symmetry axis [42] to enable realistic rendering of actors from all viewpoints.

AD Models: We evaluate two current SotA end-to-end driving models, namely UniAD [22] and VAD [23], according to our proposed evaluation protocol. In both cases, we make use of the pre-trained weights made available by the authors, trained on the same dataset, without any alterations to the configuration of said models. Both of these models consume 360° camera input, along with can-bus signals and a high-level command: *right*, *left*, or *straight*, and output a sequence of future waypoints up to 3 s into the future. While this is shorter than the initial time-to-collision (TTC) in our scenarios, it is not an issue as the evasive maneuver can, and should, begin before the final waypoint intersects the current actor position. Additionally, our scenarios are designed to be quite lenient, so that a plan at TTC < 3 s can still successfully avoid collision.

One major difference between these two models is that UniAD applies a collision-avoidance optimization post-processing step to their predicted trajectory. The optimization is performed using a classical solver with a cost-function based on predicted occupancy and the non-optimized output trajectory. This optimization was shown to drastically decrease the collision-rate when evaluated in open loop, and we can now study it in the more interesting closed-loop setting. To enable more directly comparable analysis, we implement the same collision

Table 1. NeuroNCAP evaluation results. End-to-end planners fail in novel, critical scenarios. Trajectory post-processing, as proposed in UniAD, helps significantly. The naïve baseline uses the perception of either UniAD (U) or VAD (V) to determine braking. †Corresponds to the model's original setting.

Model	Post-proc.	NeuroNCAP Score ↑				Collision rate (%) ↓			
		Avg.	Stat.	Frontal	Side	Avg.	Stat.	Frontal	Side
Base-U	–	2.65	4.72	1.80	1.43	69.90	9.60	100.00	100.00
Base-V	–	2.67	4.82	1.85	1.32	68.70	6.00	100.00	100.00
UniAD	x	0.73	0.84	0.10	1.26	88.60	87.80	98.40	79.60
VAD†	x	0.66	0.47	0.04	1.45	92.50	96.20	99.60	81.60
UniAD†	✓	1.84	3.54	0.66	1.33	68.70	34.80	92.40	78.80
VAD	✓	2.75	3.77	1.44	3.05	50.70	28.70	73.60	49.80

avoidance optimization for VAD. However, as VAD does not directly predict future occupancy, we rasterize their predicted future objects and use this as the future occupancy. Note that this approach possibly overestimates occupancy, as all future modes are treated as equally likely.

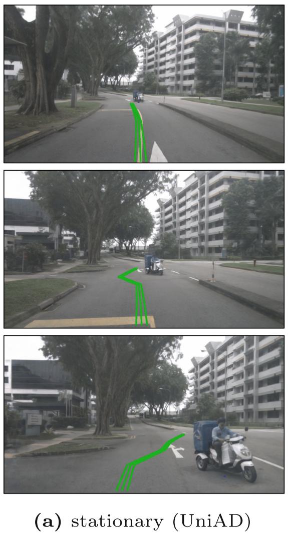
For comparison we implement a naïve baseline method based on the perception outputs of UniAD/VAD. The planning logic is simply a constant velocity model unless we observe an object in a corridor in front of the ego-vehicle, in which case we perform a braking maneuver. The corridor is defined as ± 2 meters in the lateral direction and ranging from 0 to $2v_{ego}$ meters in the longitudinal direction, *i.e.* we brake if we have TTC $< 2\,$s with an object in front of us.

4.2 NeuroNCAP Results

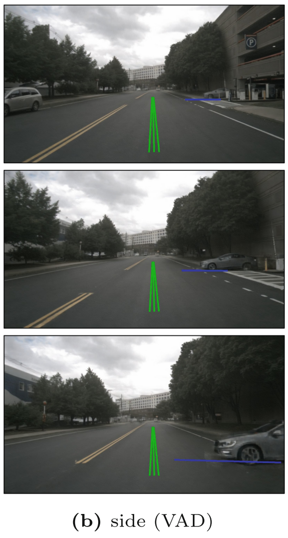
We evaluate VAD [23], UniAD [22], and the naïve baseline on our safety-critical scenarios. We also evaluate both methods with and without perception-based trajectory post-processing. We report the NeuroNCAP score (2) and collision rate per scenario type in Table 1. Note that the collision rate is not averaged over time, but is defined as the ratio of scenarios that passed without any collisions.

Surprisingly, we find that the plan predicted directly by the network, *i.e.* without post-processing, is extremely unsafe and crashes most of the time, even in the simple stationary scenarios. For reference, the naïve baseline achieves an almost perfect score in the stationary setting, showing both that the perception of these models is not at fault, and that very simple logic can avoid collision. Trajectory post-processing further confirms this, reducing the collision rate dramatically in the stationary setting. Side and frontal scenarios are more difficult to handle with this rule-based logic, and the baseline crashes 100% of the time, albeit with a lower impact speed (thus scoring higher). Surprisingly, the end-to-end methods again show almost no reaction to the impending collision, with 98–99% collision rate in frontal scenarios. Trajectory post-processing improves safety somewhat, but is not nearly as effective as in the stationary setting.

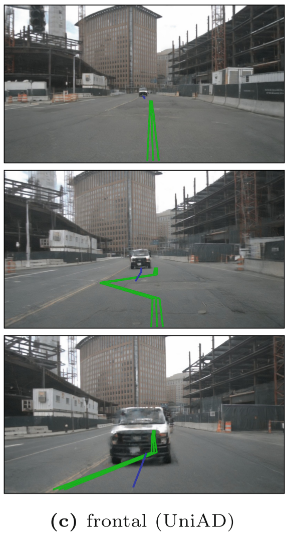
We believe that these results highlight a drastic flaw in the design or training of current end-to-end autonomous driving systems. Reducing the contradictions between the predicted plan and the auxiliary outputs is a promising area of



(a) stationary (UniAD)



(b) side (VAD)



(c) frontal (UniAD)

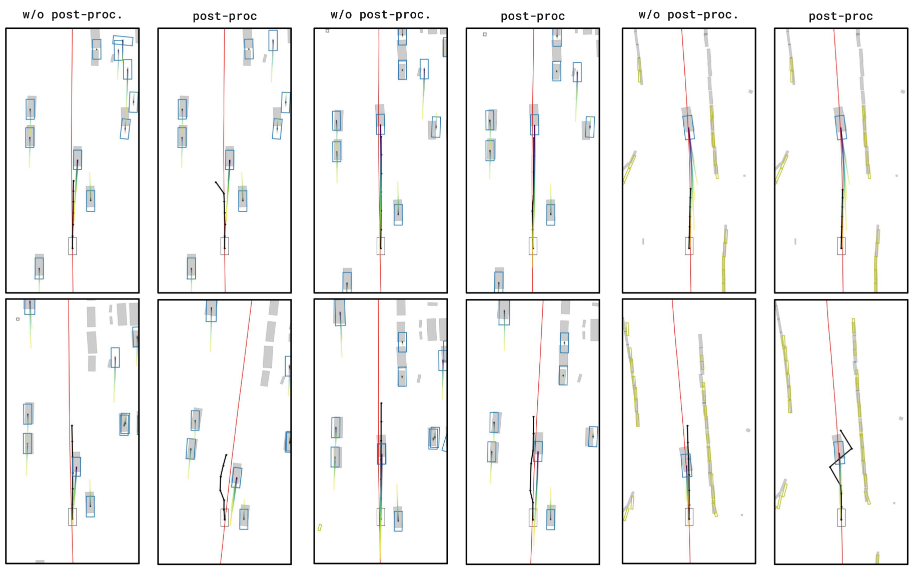
Fig. 4. Qualitative examples of three NeuroNCAP scenarios, with projected planning output (green, before controller) and the actual designed future trajectory of the target actor (blue). In some cases the planner reacts successfully (a), does not react at all (b), or attempts to avoid collision but fails (c). Our simulator can accurately render complex actors (a), but sometimes exhibits unrealistic artifacts for close objects (b) and (c). (Color figure online)

improvement for future end-to-end planners. Notably, VAD actually attempts to address this by using multiple loss terms that directly encourage the model to output a plan that is consistent with its perception and prediction outputs. However, as our experiments show, this alignment step does not generalize well, at least not to this type of safety-critical scenario.

4.3 Qualitative Results

We augment the quantitative analysis with rendered front-camera images from each scenario type in Fig. 4, with overlaid projections of the planned trajectories. Figure 4a depicts a successful avoidance maneuver, while also highlighting our ability to render complex entities such as a motorcyclist. However, without post-processing, the planners are prone to seemingly ignoring the safety-critical event, as seen in Fig. 4b.

To further examine this issue, Fig. 5 presents the perception and planning outputs across various scenarios, demonstrating that UniAD, without post-processing, frequently plans hazardous trajectories despite robust perception capabilities (left). This indicates that the high collision rate is not due to a real-to-sim gap induced by our renderer, but rather due to the planner not handling the covariate shift between the training data and our safety-critical scenarios. When post-processing is enabled (right), trajectories are adjusted to avoid collisions according to the model's internal perception and prediction outputs.



(a) Successful avoidance **(b)** Insufficient margin **(c)** Catastrophic failure

Fig. 5. UniAD perception and planning output for three different scenarios, with (right) and without (left) trajectory post-processing. Highlighting unsafe planning despite strong perception, as well as strengths and weaknesses of post-processing. The plot features ground truth objects (grey) and predicted objects (class dependent color), and their predicted future trajectories. Moreover, we show the ego-vehicle (black), its planned trajectory (black) and the reference trajectory it is steering towards (red). (Color figure online)

This can result in a few different outcomes. In some cases, such as in Fig. 5a, this is able to completely prevent a collision by slowing down and/or steering around the target. However, the optimization does not adequately consider the extent of the ego-vehicle, sometimes causing the adjustments to be too small, resulting in glancing collisions as can be seen in Fig. 5b. Finally, due to not considering the trajectory holistically, and only optimizing each waypoint individually, the result is sometimes catastrophic, as seen in Fig. 5c and Fig. 4c. Here, the waypoints are repelled from the object in the opposite direction, actually causing the vehicle to steer into the target actor and accelerate right before impact.

4.4 Simulation Gap Study

One important—if not the most important—concern of doing testing in simulation is to what degree the results transfer to the real world. Therefore, we measure the real-to-sim gap across three crucial facets of the driving system—perception, prediction, and planning—in both open and closed-loop settings.

Replay Real-to-Sim (Open-Loop): The authors of NeuRAD [37] show that the perception gap between rendered and real images is very small, at least

Table 2. Real-to-sim evaluation in open loop on our 14 nuScenes sequences, using typical planning and detection metrics. *Computed over VAD's range for fair comparison between models.

Model	Modality	ADE @ T (m) ↓			CR @ T (%) ↓			Detection ↑
		1.00 s	2.0 s	3.0 s	1.00 s	2.0 s	3.0 s	NDS*
UniAD	original	0.44	0.75	1.16	0.00	0.12	0.21	0.490
UniAD	simulated	0.47	0.80	1.24	0.00	0.12	0.24	0.489
VAD	original	0.43	0.71	1.01	0.00	0.08	0.11	0.449
VAD	simulated	0.44	0.76	1.16	0.00	0.00	0.08	0.413

Table 3. Target actor recall in different ranges and at different times in the future. Note that the model consistently has a good understanding of the target actors' dynamics as shown by the high recall during future prediction.

Scenario	Model	Recall @ 0 s ↑			Recall @ 1 s↑			Recall @ 2 s↑			Recall @ 3 s ↑		
		5–15 m	15–25 m	25–35 m	5–15 m	15–25 m	25–35 m	5–15 m	15–25 m	25–35 m	5–15 m	15–25 m	25–35 m
Stationary	UniAD	0.97	0.98	0.89	0.94	0.95	0.84	0.94	0.93	0.76	0.94	0.90	0.69
	VAD	1.00	0.96	0.70	0.97	0.87	0.64	0.93	0.82	0.60	0.91	0.80	0.57
Frontal	UniAD	0.83	0.97	0.90	0.82	0.96	0.90	0.80	0.93	0.83	0.77	0.91	0.73
	VAD	0.96	0.97	0.70	0.91	0.93	0.69	0.83	0.86	0.63	0.65	0.69	0.54
Side	UniAD	0.92	0.96	0.65	0.92	0.96	0.65	0.90	0.95	0.62	0.72	0.63	0.56
	VAD	0.90	0.64	0.44	0.87	0.64	0.40	0.82	0.62	0.38	0.56	0.51	0.36

in terms of 3D object detection and relative depth estimation. However, this study was only partially performed on nuScenes, and crucially, did not consider planning metrics, which is what is most relevant in this work. Therefore, we analyze the open-loop real-to-sim gap of VAD and UniAD on the full sequences that our scenarios are based on. In Table 2, we report the standard planning metrics Average Displacement Error (ADE) and Collision Rate (CR) metrics, computed over 1, 2, and 3 s into the future. We also present the perception real-to-sim gap in terms of NDS, utilizing the auxiliary detection outputs from each model. Our analysis is limited to VAD's maximum range of $\pm$30m and excludes pedestrians as they are not utilized in any of our scenarios (and explicitly not modeled by our renderer).

The displacement error is very similar between original and rendered images, indicating a minimal real-to-sim gap in the open-loop setting. In terms of collision rate, both VAD and UniAD show similar or lower collision rates on rendered images. In terms of detection performance, UniAD shows practically identical performance on real and simulated data, whereas VAD has a gap of roughly 3 points in terms of NDS. However, we argue that this is within acceptable margins, especially as most evaluated objects are farther away and thus less visible than our inserted target actors.

Scenario Real-to-Sim (Closed-Loop): The previous study was performed on the original sequences, where ground truth is readily available for perception, prediction, and planning. However, we also study the behavior of the models in our closed-loop scenarios, where we have edited the actors and can observe the scene from novel views. Particularly, we aim to verify that the model is able to accurately perceive and predict the motion of the inserted actor that is causing the criticality of the scenario. Thus, we evaluate metrics related to the estimation of these actors, considering recall at present (0 s) for perception and recall at future times for prediction. In calculating recall, we define a true positive as a detection or future prediction with either a non-zero overlap or a center distance smaller than 2 m with the target actor. This approach is deliberately lenient to distinguish inherent perception uncertainty in 3D estimation from failures induced by rendering artifacts. We compute and average these metrics for all our scenarios and present them on a per-range-to-actor basis in Table 3.

The results in Table 3 indicate that the model in most cases has a good understanding of the current and future states of the target actor. With high recall scores (>80%) in the most safety-critical ranges (5–25 m to the target-actor), the planner should be able to plan a safe collision avoidance maneuver. Note that VAD has a range of ±30 meters in the longitudinal direction and ±15 meters in the lateral direction. This can be seen in the overall decreased recall rate in the 25–35 m range, and in the 15–25 m range in the side scenarios.

5 Limitations

We see the following limitations. First, the neural renderer is limited in the scenes and scenarios, *e.g.*, no rain, it is able to accurately render. Moreover, large deviations in ego-vehicle trajectory and very close objects lead to visual artifacts (see Fig. 4). Second, we adopt a simplified vehicle model, which does not model, *e.g.*, delays, friction, or suspension. Further, we do not consider road surface aspects such as bumps, potholes, gravel, etc. Third, we have adopted a single controller for all models, even though they are tightly coupled. Our evaluation protocol allows for submitting AD models that directly output control signals. Fourth, the neural renderer is unable to deal with deformable objects, such as pedestrians. We hope that further advances in neural rendering will lift this restriction and enable a new set of safety-critical scenarios focusing on vulnerable road users. Fifth, the target actor follows a predetermined trajectory, without dynamically reacting to the ego-vehicle. While this follows the Euro NCAP setting, we believe that future scenarios with multiple actors would require reactive behavior.

6 Conclusion

In conclusion, our simulation environment offers a novel approach for evaluating the safety of autonomous driving models, drawing on real-world sensor data and Euro NCAP-inspired safety protocols. Through the NeuroNCAP framework, which includes stationary, frontal, and side collision scenarios, we have exposed significant vulnerabilities in current SotA planners. These findings not only underline the urgent need for advancements in the safety of end-to-end planners but also suggest promising paths for future research. By making our evaluation suite openly available to the wider research community, we aim to catalyze progress towards safer autonomous driving. Looking ahead, we anticipate evolving the suite to tackle a wider range of scenarios, integrating more refined vehicle models, and employing advanced neural rendering techniques, thereby setting new benchmarks for safety evaluation.

Acknowledgements. We thank Georg Hess, Carl Lindström, and Adam Lilja for fruitful discussions. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. Computational resources were provided by NAISS at NSC Berzelius, partially funded by the Swedish Research Council, grant agreement no. 2022-06725.

References

1. Alibeigi, M., et al.: Zenseact open dataset: a large-scale and diverse multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 20178–20188 (2023)
2. Althoff, M., Koschi, M., Manzinger, S.: Commonroad: composable benchmarks for motion planning on roads. In: 2017 IEEE Intelligent Vehicles Symposium (IV), pp. 719–726. IEEE (2017)
3. Amini, A., et al.: Learning robust control policies for end-to-end autonomous driving from data-driven simulation. IEEE Robot. Automat. Lett. **5**(2), 1143–1150 (2020)
4. Bojarski, M., et al.: End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316 (2016)
5. Caesar, H., et al.: nuScenes: a multimodal dataset for autonomous driving. In: CVPR, pp. 11621–11631 (2020)
6. Caesar, H., et al.: nuPlan: a closed-loop ml-based planning benchmark for autonomous vehicles. In: Computer Vision and Pattern Recognition (CVPR) ADP3 workshop (2021)
7. Chen, C., Seff, A., Kornhauser, A., Xiao, J.: Deepdriving: learning affordance for direct perception in autonomous driving. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2722–2730 (2015)
8. Codevilla, F., López, A.M., Koltun, V., Dosovitskiy, A.: On offline evaluation of vision-based driving models. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11219, pp. 246–262. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01267-0_15
9. Codevilla, F., Müller, M., López, A., Koltun, V., Dosovitskiy, A.: End-to-end driving via conditional imitation learning. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 4693–4700. IEEE (2018)
10. Codevilla, F., Santana, E., López, A.M., Gaidon, A.: Exploring the limitations of behavior cloning for autonomous driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9329–9338 (2019)
11. Dauner, D., Hallgarten, M., Geiger, A., Chitta, K.: Parting with misconceptions about learning-based vehicle motion planning. In: 7th Annual Conference on Robot Learning (2023). https://openreview.net/forum?id=o82EXEK5hu6
12. Deo, N., Wolff, E., Beijbom, O.: Multimodal trajectory prediction conditioned on lane-graph traversals. In: Conference on Robot Learning, pp. 203–212. PMLR (2022)

13. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: Carla: an open urban driving simulator. In: Conference on Robot Learning, pp. 1–16. PMLR (2017)
14. EuroNCAP: Assessment Protocol - Safety Assist - Collision Avoidance (2023). https://www.euroncap.com/media/79866/euro-ncap-assessment-protocol-sa-collision-avoidance-v104.pdf
15. Gao, J., et al.: Vectornet: encoding HD maps and agent dynamics from vectorized representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11525–11533 (2020)
16. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the Kitti vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3354–3361. IEEE (2012)
17. Gulino, C., et al.: Waymax: an accelerated, data-driven simulator for large-scale autonomous driving research. Adv. Neural Inf. Process. Syst. **36** (2024)
18. Hafner, D., Lillicrap, T., Ba, J., Norouzi, M.: Dream to control: learning behaviors by latent imagination. In: International Conference on Learning Representations (2019)
19. Hershman, L.L.: The us new car assessment program (NCAP): past, present and future (2001)
20. Hu, A., et al.: Gaia-1: a generative world model for autonomous driving. arXiv preprint arXiv:2309.17080 (2023)
21. Hu, S., Chen, L., Wu, P., Li, H., Yan, J., Tao, D.: ST-P3: end-to-end vision-based autonomous driving via spatial-temporal feature learning. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) ECCV 2022, Part XXXVIII, pp. 533–549. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-19839-7_31
22. Hu, Y., et al.: Planning-oriented autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 17853–17862 (2023)
23. Jiang, B., et al.: Vad: vectorized scene representation for efficient autonomous driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 8340–8350 (2023)
24. Karnchanachari, N., et al.: Towards learning-based planning: the nuPlan benchmark for real-world autonomous driving. In: International Conference on Robotics and Automation (ICRA) (2024)
25. Krajzewicz, D.: Traffic simulation with sumo–simulation of urban mobility. Fundamentals of Traffic Simulation, pp. 269–293 (2010)
26. Kudarauskas, N.: Analysis of emergency braking of a vehicle. Transport **22**(3), 154–159 (2007)
27. Liang, M., et al.: Learning lane graph representations for motion forecasting. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12347, pp. 541–556. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58536-5_32
28. Mei, J., et al.: Waymo open dataset: panoramic video panoptic segmentation. In: European Conference on Computer Vision, pp. 53–72. Springer (2022)
29. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: representing scenes as neural radiance fields for view synthesis. Commun. ACM **65**(1), 99–106 (2021)
30. Mobileye. Mobileye Under the Hood (2024). https://www.mobileye.com/ces-2024/
31. Ost, J., Mannan, F., Thuerey, N., Knodt, J., Heide, F.: Neural scene graphs for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2856–2865 (2021)

32. Pomerleau, D.A.: Alvinn: an autonomous land vehicle in a neural network. Adv. Neural Inf. Process. Syst. **1** (1988)
33. Prakash, A., Chitta, K., Geiger, A.: Multi-modal fusion transformer for end-to-end autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7077–7087 (2021)
34. Rajamani, R.: Lateral vehicle dynamics. In: Vehicle Dynamics and Control, pp. 15–46. Springer, Boston (2012). https://doi.org/10.1007/978-1-4614-1433-9_2
35. Shah, S., Dey, D., Lovett, C., Kapoor, A.: AirSim: high-fidelity visual and physical simulation for autonomous vehicles. In: Hutter, M., Siegwart, R. (eds.) Field and Service Robotics. SPAR, vol. 5, pp. 621–635. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-67361-5_40
36. Son, T.D., Bhave, A., Van der Auweraer, H.: Simulation-based testing framework for autonomous driving development. In: 2019 IEEE International Conference on Mechatronics (ICM), vol. 1, pp. 576–583. IEEE (2019)
37. Tonderski, A., Lindström, C., Hess, G., Ljungbergh, W., Svensson, L., Petersson, C.: Neurad: neural rendering for autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2024)
38. Van Ratingen, M., et al.: The European new car assessment Programme: a historical review. Chin. J. Traumatol. **19**(02), 63–69 (2016)
39. Watter, M., Springenberg, J., Boedecker, J., Riedmiller, M.: Embed to control: a locally linear latent dynamics model for control from raw images. Adv. Neural Inf. Process. Syst. **28** (2015)
40. Wilson, B., et al.: Argoverse 2: next generation datasets for self-driving perception and forecasting. In: Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021) (2021)
41. Xiao, P., et al.: Pandaset: advanced sensor suite dataset for autonomous driving. In: 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), pp. 3095–3101. IEEE (2021)
42. Yang, Z., et al.: Unisim: a neural closed-loop sensor simulator. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1389–1399 (2023)