

Automated Generation of Dynamic Occlusion-Caused Collisions

by

Eli-Henry Dykhne

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2024

© Eli-Henry Dykhne 2024

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

This research was performed as a collaboration between members of the Waterloo Intelligent Systems Engineering Lab. Eli-Henry Dykhne designed and implemented the routines that performed occlusion detection, the notice system, scenario generation, the NPSC (NuPlan Scenario Collection) file format, and additional metric tracking. He was also responsible for running experiments and analyzing the results. Shea Anthony Cardozo was responsible for implementing the various manager classes that allowed non-ego vehicles to be controlled by various planners. James Riddell, was vital in the initial prototyping phase of the occlusion insertion system. Professor Krzysztof Czarnecki developed the dynamic occlusion typology.

Abstract

Dynamic occlusions (occlusions caused by other moving road objects) pose some of the most difficult challenges for autonomous driving systems (ADS). Validating the robustness of ADSs to safety-critical dynamic occlusions is a difficult task due to the rarity of such scenarios in recorded driving logs. We provide a novel typology of dynamic occlusion scenarios involving vehicles, as well as a framework for ADS safety validation in the presence of dynamic occlusions. Our framework allows for the generation of a diverse set of dynamic occlusion-caused collisions (OCCs) across a wide variety of intersections. We provide results demonstrating that our technique achieves higher generation efficiency and diversity of OCCs than prior works while being applicable to a wide range of intersections. In this work, we present our generation technique and provide a detailed analysis of the variety and quality of our generated scenarios, as measured by typology coverage, scenario severity, and robustness to faster reaction times.

Acknowledgements

This work was greatly helped thanks to the contributions of Shea Anthony Cardozo. His help with the modifications to our simulator significantly were vital to the prototyping phase of our research. My supervisor, Professor Krzysztof Czarnecki has been a steady guide throughout this research, and an invaluable source of information that directly shaped this work.

Vahdat Abdelzad, James Riddell, Professor Sean Sedwards and all the others in the Waterloo Intelligent Systems Engineering (WISE) lab were vital in keeping me on the right path throughout my research. I would also like to specifically thank Luke Rowe, who, in my first year, took me under his wing and helped me settle into my research.

Dedication

This is dedicated to my mother, Marina, my father, Leonid and my grandmother, Larissa, who all showed unwavering support for me as I did my research, making sure I had everything I needed so that I could focus on the work. I would also like to dedicate this to my friends, who made sure I always felt like I could lean on them for support. Even 100 kilometers away, I never felt alone.

Table of Contents

List of Figures	ix
List of Tables	xi
1 Introduction	1
2 Literature Review	4
2.1 Simulation Environments	4
2.2 Scenario Generation For ADS Validation	5
2.2.1 Adversarial Methods	5
2.2.2 Agent Insertion and Placement	6
2.2.3 Scenario Variety	7
2.3 Occlusion Scenario Generation	7
3 Dynamic Occlusion	9
3.1 Defining Dynamic Occlusions	9
3.2 Dynamic Occlusion Typology	10
3.2.1 Vehicle to Vehicle Conflict Types	10
3.2.2 Movement of Occluding Vehicle	11

4	Method	13
4.1	NuPlan	14
4.2	OCC-Gen	15
4.3	Base Scenario Selection	16
4.4	Control	17
4.5	Occlusion Computation	17
4.6	Reaction Time, Evidence Accumulation, and Attentiveness	18
4.7	Scenario Generation	21
4.7.1	Cross Conflict Type Generation	22
4.7.2	Cross and Merge Conflict Type Generation	23
4.7.3	Sequential Conflict Type Generation	24
4.7.4	Diverge Conflict Type Generation	24
5	Experiments and Evaluation	27
5.1	Typology Coverage	27
5.2	Intersection Variety	29
5.3	Generation Efficiency	30
5.4	Severity Analysis	31
5.5	Lone Occluder Insertion Ablation	36
5.6	Reaction Time Ablations	36
5.6.1	Ego Reaction Time	37
5.6.2	All Agent Reaction Time	37
6	Threats to Validity and Future Work	39
6.1	Threats to Validity	39
6.2	Future Work	40
7	Conclusion	41
	References	42

List of Figures

1.1	An example scenario generated using our technique and simulated in the nuPlan simulator [5], showing an occlusion-caused collision. The white vehicle, turning left, and the black vehicle, going straight, are occluded by the green vehicle as it makes a left turn, leading to a head-on collision as they fail to see each other in time.	2
4.1	The high level framework used to generate our scenarios. We use a dual filter technique to first remove scenarios where no collision occurs, and then remove scenarios where the collision is avoided by the planner when it is given information that would have otherwise been hidden by the occlusion.	14
4.2	The steps taken to detect which objects are occluded. (a) The agents are processed nearest to furthest. (b) The wedge pointing to the centre of the current agent is found. (c) Once the angular diameter of the agent is found, (d) the number of wedges to check to either side of the wedge pointing to the centre of the agent is determined. We continue with the next step even if the initial wedge is no longer in our set. If enough wedges intersect the agent, it is determined to be visible. (e) Wedges that intersect our agent are removed from the overall set of wedges, meaning that objects in the resultant shadow will be occluded.	20

4.3	The view of the present and past that the motion planner is given of this object. OHB refers to “Observation History Buffer”, provided to the planner. The shaded blue region visually displays the uncloak window. This example demonstrates the notice system with a 5-frame uncloak window and a 3-frame notice threshold. The object only becomes visible in the observation history at time-step 8, meeting both conditions of 3 frames of visibility and being visible at the start of the notice window. This shows that an object can be noticed while invisible (e.g., step 5 during time-step 8) and can be noticed on an invisible step (e.g., time-step 8). Previously unnoticed time-steps can become noticed once the system processes the appearance, revealing the trajectory for all steps that are both visible and noticed. . . .	22
4.4	The steps taken to generate an OCC using the cross conflict type generator. Ego is grey, other vehicles are green. Lanes are in blue, lane connectors are red. Planned paths are denoted by arrows.	25
5.1	Radial plot of impact angles and ΔV for each generator. Impact angles are relative to ego’s heading when ego collides with another vehicle in an OCC.	33
5.2	Severity class distributions of different types of OCC	35

List of Tables

3.1	Dynamic Occlusion Patterns. Single, non-exhaustive examples per category.	10
5.1	Generation coverage of the typology. Blue circles signify types that the most closely related prior work [24] was also capable of generating.	28
5.2	Generation coverage of the typology on a signalized two-lane cross intersection.	29
5.3	Unique Intersection Generation	30
5.4	Generation Efficiency	31
5.5	The AIS injury scale categorizes severity from 1 (minor) to 6 (fatal) [9] . .	34
5.6	Classification of Damage Based on Safety-Related Probability[52]	34
5.7	Collision Severity Categorization by Impact Side and ΔV Based on Conservative Intervals from [52]	35
5.8	Percent of OCCs Still Occurring After Varying Ego Reaction Delay (Non-ego reactions set to (1.5, 1.0))	37
5.9	Percent of OCCs Still Occurring After Varying Agent Reaction Delay (ego reactions set to (0.5, 0.33))	38

Chapter 1

Introduction

Consider the developer tasked with designing an autonomous driving system (ADS). Their top priority is the safety of their driving software. To ensure the ADS can safely handle hazardous on-road situations, the developer may choose to test the system in simulation before testing it on the closed course and later deploying it on real roads. In this simulated environment, the developer can evaluate the ADS against a wide range of scenarios, including both recorded real-world events and custom-designed scenarios aimed at probing specific vulnerabilities. However, if a critical scenario type is both rare and highly variable, it becomes challenging to accumulate enough examples to adequately cover all possible configurations this scenario might take across different roadways and traffic conditions. In such cases, automatic scenario generation becomes essential for producing the variety needed to ensure comprehensive safety validation. Critical *dynamic occlusions* (cases where collisions are caused by a moving object that obstructs the sight of a driver) are one example of such rare yet highly variable scenarios, and they remain largely under-addressed by existing scenario generation software.

The difficulty of collecting and testing dangerous and unusual road scenarios to achieve a high test coverage remains a limiting factor in validating autonomous driving systems. Prior work has mostly focused on testing the ADS through simulated replay of real-world scenarios from on-road datasets captured by data collection vehicles [5, 38], as well as testing on manually designed scenarios [2]. This has led to significant research in automated critical scenario generation, where the behaviour of ADSs can be verified against adversarial agents [48]. However, comparatively little research has focused on generating critical scenarios caused by occlusions, which are responsible for 11.6% of collisions [42]. The scarcity of scenarios is even more acute in the area of dangerous dynamic occlusions, because while dynamic occlusions are not rare on the road, scenarios where they cause

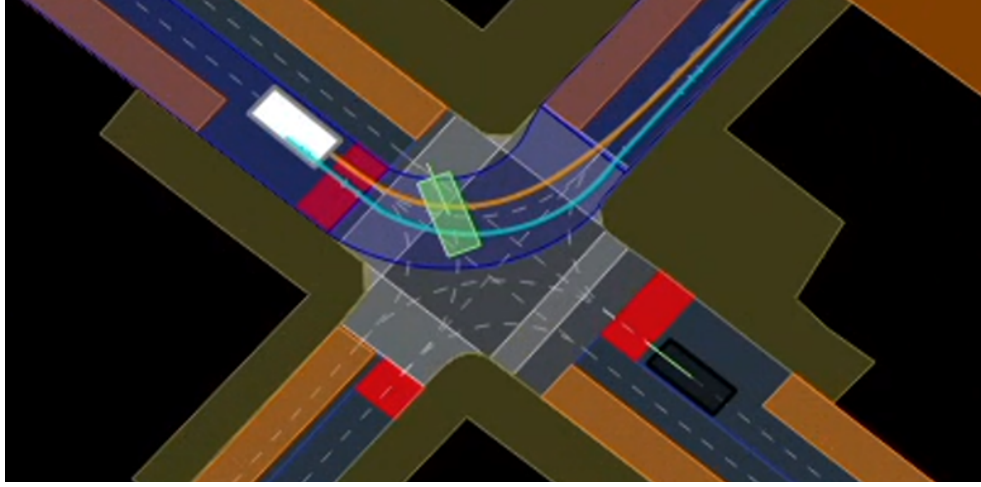


Figure 1.1: An example scenario generated using our technique and simulated in the nuPlan simulator [5], showing an occlusion-caused collision. The white vehicle, turning left, and the black vehicle, going straight, are occluded by the green vehicle as it makes a left turn, leading to a head-on collision as they fail to see each other in time.

collisions are. The potential variety of such scenarios also poses a problem for validation, as any scenario where three vehicles fall on a line where two of them are on intersecting paths, could potentially result in an occlusion-caused collision. Prior works have focused on particular types of OCCs [1, 24, 39, 56, 60] but make no attempt to capture the complete range of possible scenarios. The generation of dangerous occlusion scenarios for the validation and training augmentation of ADSs across a variety of roadways is vital to assuring the safety of such systems.

Broadly, the research contributions of this thesis are:

- A novel typology for categorizing dynamic occlusions across a variety of road geometries.
- A set of techniques for generating dangerous dynamic occlusions, which can then be filtered for occlusion caused collisions, such as the one displayed by Figure 1.1.
- Empirical analysis of the severity, variety, robustness to reaction time, and coverage of the generated dynamic occlusion caused collisions.
- An open source tool for the generation and analysis of occlusion caused collisions.

The remainder of the thesis is organized as follows.

- Chapter 2 provides a literature review of occlusion related research, including generation and planning.
- Chapter 3 is adapted from [17]. It defines the concept of a dynamic occlusion and describes the dynamic occlusion typology.
- Chapter 4 is adapted from [17]. It explains the methodology behind how the agents' behaviors within the simulation are controlled, how the occlusion computation and notice system work, and how the scenario generators function.
- Chapter 5 is adapted from [17]. It details our experiments and results. Here, we address our questions regarding the efficiency of our generation method, the variety of scenario types produced, the range of intersections where scenarios can be generated, the robustness of our method to drivers with better reaction times, and the severity of the generated scenarios.
- Chapter 6 examines the factors that could impact the validity of our findings.
- Chapter 7 serves to sum up our contributions and results.

Chapter 2

Literature Review

Scenario based testing has been thoroughly explored as a method for ADS validation [49]. This method allows the system under test (SUT) to demonstrate that it can safely carry out all the tested scenarios. To achieve a sufficient coverage of the expected scenarios that the SUT has to operate in, the scenarios being tested on must be diverse and also include potentially rare but critical scenarios that may come up on the road. Test suites that can cover all critical scenarios are difficult to create, as they exist in an extremely large potential space of realistic scenarios. A data collection vehicle may drive for thousands of hours without encountering a single critical scenario, making reliance on collected scenarios infeasible. This review is primarily split into general scenario generation and occlusion scenario generation.

2.1 Simulation Environments

The field of ADS validation relies heavily on simulation. As mentioned in several surveys on the subject [6, 14], there exist a number of simulation and testing tools that are commonly used in research.

- CARLA [15]: a high fidelity traffic simulator with its own maps, and tools for generating new maps. It can be used for testing and agent training.
- SUMO [32]: a large scale, low fidelity traffic simulation tool. It is mainly used for traffic flow simulation.

- SMARTS Scenario Studio [73]: a low fidelity simulation platform for multi-agent reinforcement learning.
- CommonRoad [2]: a low fidelity simulator and an environment for the construction of new scenarios as well as the training of agents and benchmarking.
- nuPlan [5]: a low fidelity simulator originally built for benchmarking that is packaged with four detailed maps and a large database of labeled scenarios that were recorded in real traffic.

We select nuPlan due to its large database of real recorded traffic scenarios and multiple alternative smart agent models available for simulating vehicle behavior.

2.2 Scenario Generation For ADS Validation

Scenario generation for ADS testing has an extensive literature. Most existing methods are generic and not oriented towards specific classes of critical scenarios. In general, scenario generation methods can be classified into coverage-oriented and criticality-oriented [7]. Coverage-oriented methods include random sampling, combinatorial testing, and mutation testing. Criticality-oriented methods focus on critical scenarios, such as finding high-severity or high-risk collision scenarios, and include various sampling and optimization methods like importance sampling, heuristic search, reinforcement learning, and boundary-focused search. As part of the optimization and search, different aspects of a scenario can be modified, including road structure, number of vehicles, position of vehicles, vehicle trajectories, vehicle behaviors, vehicle goals, and visibility. The method presented in this thesis addresses both categories: it is coverage-oriented by being driven by a scenario typology and criticality-oriented by targeting a specific class of dangerous scenarios.

The remainder of this section describes several state-of-the art criticality-oriented methods that use some form of optimization, also known as adversarial methods; methods for inserting agents into driving scenarios; and the issue of achieving a high scenario variety during generation.

2.2.1 Adversarial Methods

A common approach to generating critical scenarios involves altering the behaviors of non-ego vehicles to make them more aggressive. These methods are classified as adversarial

since the vehicles in these scenarios actively seek to push the System Under Test (SUT) into critical situations. Several works [21, 23, 40, 51, 64, 69] directly optimize agent behavior parameters, such as braking, acceleration limits, and thresholds for minimum acceptable safety distances, to induce these scenarios. For example, J. Ge et al. [22] tune a *politeness* parameter in MOBIL [25], which dictates the degree of consideration for surrounding vehicles during lane changes.

Other works [58, 68] employ reinforcement learning, using reward functions that encourage non-ego vehicles to behave dangerously around the ego vehicle. Common reward mechanisms include considerations for collisions, time to collision (TTC), and realism metrics such as road rule compliance and deviations from ground truth trajectories.

Adversarial approaches must carefully balance the creation of critical scenarios with the need for agents to behave in a reasonable manner. An exception is the work of Y. Yang et al. [65], which addresses a special case involving a suicidal pedestrian running onto the street. Here, the vehicle is expected to stop in time, if possible. Similar to our work, this approach involves inserting conflict agents into the scene to create critical scenarios. However, unlike our method, these agents are trained to act in an adversarial manner.

Finally, Rempe et al. [48] also utilize an adversarial approach. To ensure realistic vehicle behavior, they use a conditional variational autoencoder to generate plausible driver trajectories, which are then optimized to create dangerous situations while staying within the realm of likely trajectories. A subsequent optimization stage tests if it is possible to refine the ego vehicle’s trajectory to enable it to escape the collision, thereby retaining only solvable scenarios for the SUT.

In contrast to adversarial methods, which leverage optimization, our method is heuristic, leveraging domain knowledge to place and initialize non-ego vehicles such as to create critical situations. We leave the optimization of our method, such as optimizing placement, initialization, and agent behavior to future work.

2.2.2 Agent Insertion and Placement

Most scenario generation techniques do not modify the number of agents or their positions [29, 64, 70]; however, those that do are more closely related to our work. For instance, L. Feng et al. [29] employ an autoregressive sampling process to position vehicles one by one, based on a spatial probability distribution learned from real-life scenarios. Although this method does not necessarily generate critical scenarios, it tends to produce more realistic and varied outcomes. Similarly, Z. Wei et al. [70] formulate scenario generation as a game, enabling a reinforcement learning (RL) agent to edit scenarios to produce critical

situations. The RL agent can perturb the trajectories of existing agents, add new agents, and introduce undrivable regions to the map, such as parked vehicles or construction areas. The agent is guided by a reward system that encourages the creation of risky scenarios while maintaining realism, measured by the likelihood and feasibility of the resulting agent trajectories.

2.2.3 Scenario Variety

Many studies showcase their technique on only a single type of cut-in, cut-out or otherwise two-lane scenario with a low number of tunable parameters, often not even modifying the number of vehicles taking part [21, 22, 40, 66, 69]. Works that target a wider set of parameters, such as a diverse set of road geometries and number, positions, and behaviors of vehicles are rare (e.g., [29, 62, 63, 70]). Even if a variety of parameters can be modified, methods, such as adversarial ones, still need to be encouraged to generate a wide variety of scenario types. This requires ways to measure scenario diversity. For example, Mahadikar et al. [33] propose a similarity metric for scenarios.

In this work, we consider a wide range of road geometries, different placements and initialization of vehicles, and scenario diversity is achieved by covering different types from a scenario typology.

2.3 Occlusion Scenario Generation

Most existing literature related to occlusion addresses planning methods to handle occlusions [3, 8, 37, 41, 45, 53, 55, 59, 72]; however, only a few address occlusions in test scenario generation.

Rather than building scenarios from scratch, some methods (ours included) rely on real scenarios as a starting point. For example, AdvSim [60] converts existing scenarios to be run in a LiDAR simulator and extrapolating what occluded areas would look like, so that they can simulate LiDAR in the occluded regime. Rather than injecting new agents as in our approach, they modify the behaviour of existing agents to cause collisions in their simulation. As another example, Muslim et al. [39] is based exclusively on the cut-out scenario on the highway, which we also explore, though in the context of a diverging lane rather than a lane change. They propose a data driven technique for validating ADSs in this particular scenario, though do not actually perform the experiments. Further, Agarkar et al. [1] focuses primarily on generation of large numbers of scenarios via fuzzing of a base

scenario. It modifies weather and lighting conditions that complicate the perception task and injects static occlusions that block pedestrians to create its scenarios.

As mentioned previously, the closest prior work [24] performed similar agent injections to create OCCs. Their technique is built to work on only a single intersection. Unlike ours, they only inject the occluder and focus on creating conflicting trajectories via satisfying a Nash equilibrium in an occluded regime, rather than relying on existing planners.

Finally, the DriveOcclusionSim benchmark in [56] generates its scenarios by randomly placing vehicles, obstacles and vulnerable road users (VRUs) in the simulation. It focuses on slightly different types of occluded scenarios than us including VRUs emerging from behind parked cars red light infractions and the canonical left turn occlusion covered in both our work and [24].

Chapter 3

Dynamic Occlusion

A general traffic occlusion occurs when an object obstructs a road user from seeing another part of the road. This prevents the road user from seeing a potential threat, such as a traffic conflict or damage to the road. A traffic conflict is a situation in which two or more road users maintain trajectories such that there is a risk of collision [30]. With these definitions in place, we can move towards defining the concept of a dynamic occlusion, as well as our typology.

3.1 Defining Dynamic Occlusions

Whether an occlusion is static or dynamic depends on the nature of the occluder. Static occlusions are caused by objects that are stationary and not expected to move soon, such as trees, buildings, and parked cars. Dynamic occlusions are primarily caused by moving vehicles, or those that may move shortly. Trucks, buses, cars can occlude other cars, vulnerable road users (VRUs), and foreign object debris (FOD).

While static occlusions are localized to the blind spots generated by buildings and large obstructions, dynamic occlusions can occur anywhere on the road, making identifying and dealing with them a much harder task. For this work, we focus on dynamic occlusions where all three objects are vehicles.

3.2 Dynamic Occlusion Typology

The type of dynamic occlusion is dependent on the traffic conflict type, which stems from the relative positioning of the two conflict vehicles [10, 11, 28], as well as the positioning of the occluding vehicle relative to the two conflict vehicles. A conflict vehicle is defined as a vehicle on a collision course with ego (the focus vehicle). Traffic conflicts between vehicles occur at conflict points. Static conflict points are the results of road geometries, and occur where lanes cross, merge and diverge. Dynamic conflict points are not tied to static road geometries but instead occur due to the movements of the agents on the road, such as lane changes, jaywalking or differences in the speed of vehicles sharing the same lane.

Table 3.1: Dynamic Occlusion Patterns. Single, non-exhaustive examples per category.

Conflict Occluder	Sequential	Cross	Merge	Diverge	Head-On
Sequential	N/A				N/A
Cross					
Merge					
Diverge					
Parallel					

Legend: Arrows represent relative vehicle movements. A circled arrowhead denotes occluder; the remaining two arrowheads denote conflict vehicles.

An arrowhead terminated by a bar denotes a stopped vehicle.

3.2.1 Vehicle to Vehicle Conflict Types

Vehicle-to-vehicle conflicts reduce to the five base conflict types: Sequential, Cross, Merge, Diverge, and Head-On [10, 36]. In this work, we exclude head-on conflicts due to the absence of two-way roads in the nuPlan dataset, although our methods are adaptable

to maps containing such specialized lanes. Vehicle speeds, lane structures, and traffic control devices such as traffic lights significantly impact the likelihood and severity of these conflicts. In order to inoculate an ADS system against dynamic occlusions, it is important to cover as many types, on as large a variety of intersections as possible.

The movement type of the two vehicles can be described as follows:

1. Sequential: The two vehicles share a lane and move one after the other.
2. Cross: The two vehicles inhabit different lanes that intersect one another, but do not merge.
3. Merge: The two vehicles inhabit different lanes that merge together into a single lane.
4. Diverge: The two vehicles share a lane, but diverge from one another as the lane splits into two different lanes.
5. Head-On: The two vehicles share a lane, but move in opposite directions.

3.2.2 Movement of Occluding Vehicle

The positioning and movement of the occluding vehicle is crucial in determining the type of dynamic occlusion. Occluding vehicles can be either Sequential, Cross, Merge, Diverge, and Parallel with respect to the conflict vehicles. Depending on the geometry of the intersection, a single scenario may feature an occluder that fits multiple categories simultaneously, such as an occluder crossing the path of one agent while being parallel to another. In such cases, the scenario is classified as both types.

The full type of a given occlusion scenario consists of two parts. First, a label for the relative geometry of the lanes used by the two conflict vehicles. Second, a label for the geometry of the occluder relative to the two conflicting vehicles. For example, a typical cut-out scenario can be classified as a sequential conflict with a diverging occluder. As more agents become part of the scenarios, more complex occlusions can be created, where multiple occluders work together to create OCCs.

The movement type of the occluding vehicle can be described as follows:

1. Sequential: Occluder and vehicle share a lane and move one after the other.

2. Cross: Occluder and vehicle inhabit different lanes that intersect one another, but do not merge.
3. Merge: Occluder and vehicle inhabit different lanes that merge together into a single lane.
4. Diverge: Occluder and vehicle share a lane, but diverge from one another as the lane splits into two different lanes.
5. Parallel: Occluder and vehicle inhabit two separate lanes which are parallel to each other. They do not need to be going in the same direction.

Table 3.1 illustrates dynamic occlusion patterns with respect to combinations of conflict types and types of occluding vehicle.

Chapter 4

Method

Our method broadly follows the technique outlined in [24], where real scenarios involving vehicles, pedestrians and obstacles are captured by either external or onboard sensors and then injected with dynamic occlusions, with the potential to cause an OCC, by injecting extra vehicles into the scene. Unlike [24], where only the occluder is injected, we also inject the conflict vehicle in order to increase the likelihood of generating a collision. Also unlike [24], the injection of occluder and conflict vehicles is guided by our systematic occlusion typology. The generated scenarios are then run with occlusions enabled to see which result in a collision. This produces a subset of scenarios where collisions occur, but where the collision may or may not be caused by the presence of the occlusion. For example, the vehicles might have already been on an unavoidable collision course. This subset of scenarios is then re-run with occlusions turned off, as if all vehicles are completely see-through. Only those scenarios where the collision does not occur in the see-through regime are classified as OCCs. The result of this process is that the generated scenarios have been filtered twice as shown in Fig. 4.1, leaving only those where the occlusion is to blame for a resulting collision. Since these scenarios are generated relative to the reactions of the ego vehicle or SUT, this method is an adversarial one. The generated scenarios then are specific areas where this particular planner fails (though the same scenarios can be used to test other vehicles).

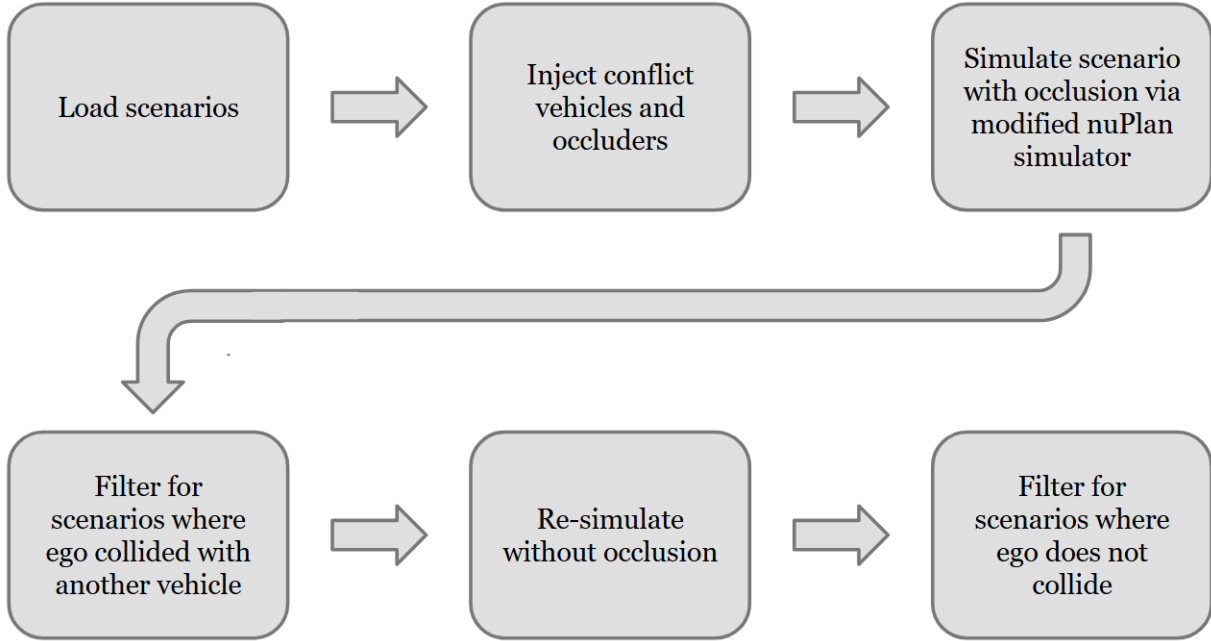


Figure 4.1: The high level framework used to generate our scenarios. We use a dual filter technique to first remove scenarios where no collision occurs, and then remove scenarios where the collision is avoided by the planner when it is given information that would have otherwise been hidden by the occlusion.

4.1 NuPlan

Our generator implementation makes use of the nuPlan simulator [5]. The nuPlan simulator is a lightweight, extensible platform for evaluating motion planners. It can operate in both open and closed loop modes, while tracking planning-specific metrics. The 2023 nuPlan Planning Challenge involved running submitted planners against three types of challenges.

1. Open-loop prediction challenges, where based on the scenario history, the planner was expected to predict the ground truth trajectory of the ego vehicle.
2. Closed-loop driving challenges against unreactive drivers, where scenarios are navigated against other road agents that move according to prerecorded trajectories. Agents are evaluated according to a number of metrics to evaluate comfort, safety, progress towards a goal and compliance with traffic rules.

3. Closed-loop driving challenges against reactive drivers controlled by simple IDM (intelligent driving model), where the agents are evaluated the same as they were in the closed-loop, unreactive regime.

In addition to adapting the simulator to our needs, our work makes use of nuPlan’s extensively labelled dataset of on-road scenarios, collected by driving a vehicle with multiple cameras and LiDAR sensors for 1500 hours across Las Vegas, Pittsburgh, Boston, and Singapore.

4.2 OCC-Gen

OCC-Gen is our modified version of the nuPlan simulator [5], in order to support occlusion simulation, injection of agents into existing scenarios to create occlusion-amplified traffic conflicts, independent behavior simulation of agents, and verification of occlusion-caused collisions. Our changes create an alternative variation of the closed-loop reactive regime to allow different types of drivers (i.e., driving behavior algorithms and models) to control non-ego vehicles. This is done for two reasons. First, the original implementation on nuPlan supported only the Intelligent Driver Model (IDM) [57] for controlling non-ego vehicles; however, the non-IDM driving agents available to us as part of nuPlan (but runnable only for ego) are significantly more capable. This is evidenced by their higher safety and progress scores on the nuPlan benchmark. As a result, using these more capable agents for non-ego vehicles will result in fewer collisions caused by poor driving decisions that are not directly related to occlusions. Second, the IDM agent as implemented in nuPlan has access to information about only the agent it is following in the same lane, and as a result already operates in a semi-occluded regime (and suffers greatly for it in critical scenarios), as it pretends that all other agents on the road would not exist. For our system to be capable of generating occlusion caused collisions, we must have a planner that can see agents which we can occlude or leave unoccluded.

In our implementation, we allow non-ego agents to be controlled by UrbanPlanner [54], IDM [5], GC_PGP [19], and the three variations of the PDM planner from tuPlan [12]. Any other planner can be integrated to control ego. Further, OCC-Gen includes our custom occlusion system, which can be extended with custom occlusion modules. It also implements our custom notice system, explained later. Finally, our tool provides a method for exporting, loading and rerunning scenarios in custom nuPlan scenario collection files (NPSC). This allows the same scenarios to be run again with different settings.

4.3 Base Scenario Selection

The scenarios we use as input to our OCC scenario generation algorithm are filtered by multiple parameters. The nuPlan dataset provides a number of options that allow us to filter by map, scenario type, recorded ego movement distances, and speeds, along with controls for how frequently to sample scenarios. For our experiments, we selected scenarios from all maps, with the restriction that ego must have traveled at least 10 m in order to filter out scenarios where ego is blocked or waiting at a red light for the entire duration of the scenario. NuPlan has 73 scenario types, including scenarios of type “unknown”. In order to maximize the probability that chosen scenarios are good candidates for modification, we select scenarios only of the following types:

1. `starting_left_turn`
2. `starting_right_turn`
3. `starting_straight_stop_sign_intersection_traversal`
4. `starting_straight_traffic_light_intersection_traversal`
5. `starting_unprotected_cross_turn`
6. `starting_unprotected_noncross_turn`
7. `traversing_intersection`
8. `traversing_traffic_light_intersection`

These were chosen specifically because of the starting proximity to intersections where ego was expected to traverse the intersection rather than stop at it. Notably, scenario types involving protected turns were excluded due to such scenarios being unlikely subject to collisions as a result of the protection. A more inclusive selection may have yielded more test scenario types, including those involving u-turns, but likely at a high computational cost. We leave exploring the remaining types for future work. The maximum number of scenarios per type was different between various scenario generators, as some types required far more recorded scenarios than others to generate a similar amount of test scenarios.

4.4 Control

For our experiments, we control the ego agent with a motion planner. Recorded pedestrians, cyclists, and non-vehicle objects are simply controlled via playback. Non-ego vehicles are controlled by a separately chosen vehicle planner, except for vehicles determined to be parked, which remain stationary. Vehicles that are both far from the ego vehicle and on non-intersecting paths, are controlled by a simple IDM controller, to reduce unnecessary computation.

In our experiments, for both the ego vehicle and all other vehicles requiring a planner, we chose the winning entry of the 2023 nuPlan challenge [5]. Specifically, we selected the PDM-Closed model [12], which performed best in the closed reactive portion of the nuPlan challenge. The PDM-Closed model is based on simple center-line following with positional and speed offsets, while simulating forward via an internal model of the nuPlan simulation environment. The trajectory that produces the highest scoring result from the heuristic metrics is then chosen.

This model was chosen to maximize the likelihood of safe driving and collision avoidance, as well as to benefit from its faster inference time compared to many heavier neural network-based alternatives. Although we selected this particular model to control our drivers, our method and tooling are capable of optimizing collisions for different planners. Ego and other drivers do not need to use the same planner.

All planners are given access to the HD map in whatever format they require (raster, vector, etc.), as well as the history of their own prior trajectory and the dimensions and trajectories of all other tracked objects. The visibility of other tracked objects in this history is determined by our occlusion computation and the notice system, which determines if the vehicle has had enough time to notice the other object.

4.5 Occlusion Computation

In order to determine if objects are visible or occluded, we implement the occlusion computation algorithm as described in Alg. 1 and visualized in Fig. 4.2. The occlusion computation treats objects, pedestrians, and cyclists as see-through while vehicles are opaque, and can thus cause other objects to become occluded. In order to increase the speed of computation, we use a lightweight proxy for ray casting by creating 720 triangular wedges in a disc with ego as the centre-point to emulate its field of view. This number results in an angular resolution of 0.5° , which is in the range typical for LiDAR sensors

used in autonomous driving [50]. For a vehicle to be visible, it must be intersected by at least 3 wedges. This allows the algorithm to simulate small pieces of cars being exposed without making the whole vehicle appear, much like a human or object detection algorithm would be likely to miss an object if only a sliver was exposed from behind another object. The algorithm works by processing the objects from nearest to furthest. It checks that the wedges that are in the sector that should cover the current object are still present, and then if the object is marked as opaque, it removes all wedges intersecting that object, such that all future checks against objects further away will be unable to check in that sector which is being occluded by this current object. This allows us to reduce the number of checks that need to be made in the future. As noted in [24], the occlusion relation is asymmetric. It is possible for one vehicle to be occluded from ego’s view, while that vehicle is still being capable of seeing ego, and vice versa.

4.6 Reaction Time, Evidence Accumulation, and Attentiveness

Neither humans nor machines can instantly react to changes in the driving environment. Humans are limited by the time it takes to process new information and then make physical braking and steering inputs. ADSs are limited by a similar processing lag, as well as the time it takes to determine if a given detection is a false positive or not. This is often done by requiring objects to be detected for multiple frames before they are confirmed as real objects and sent to the planning system to be added to its current world-view.

Researchers de Winter et al. [13] found that humans can typically react and apply brakes within 0.3 to 1.0 s, to scenes that have artificial occlusions introduced by blanking the screen of a simulator. The test subjects in this experiment were given practice runs and asked not to make the decision to brake until after the occlusion lifted, which is likely to have primed them and improved their results. Experimental research in cognitive neuroscience shows that humans accumulate evidence, such as visual stimuli, up to some threshold, before recognizing objects and making decisions [46]. Evidence accumulation has been linked to perception-reaction time in driving [18, 34], which is influenced by the accumulation rate (e.g., accumulating sightings of a highly occluded vs. fully visible object), surprise level (whether evidence matches prior expectations), and distraction level. In [4, 16, 31], drivers were measured as having reaction times between 0.5 and 1.7 s. A significant increase was found in those that compared phone distracted driving with undistracted driving. Drivers reacting to a looming vehicle (one that has braked unexpectedly while remaining in full view) [43] were observed with comparable reaction times centering on 1.04 s.

Algorithm 1: Determine Occlusions

input: *observer*: Object we calculate occlusions for.
 targets: Objects which may be occluded.
 numWedges: Controls angular resolution.
 minHits: Minimum wedge hits to be visible.

```
1 wedges  $\leftarrow$  GenWedgePolys(numWedges)
2 sortByDistance(targets, observer)
3 notOccluded  $\leftarrow$   $\emptyset$ 
4 for target  $\in$  targets do
5   wedgeHits  $\leftarrow$  0
6   targetCorners  $\leftarrow$  ConvertToObsFrame(observer, target)
7   targetPoly  $\leftarrow$  Polygon(targetCorners)
8   angle  $\leftarrow$  GetAngleToTarget(observer, target)
9   ctrWedgeIdx  $\leftarrow$  int(angle/2 $\pi$   $\times$  numWedges)
10  toRemove  $\leftarrow$   $\emptyset$ 
11  diameter  $\leftarrow$  AngularDiamObsPOV(target, observer)
12  numWedgesToCheck  $\leftarrow$  LengthToNWedges(diameter, numWedges)
13  for i  $\leftarrow$  -numWedgesToCheck/2 to numWedgesToCheck/2 do
14    wedgeIdx  $\leftarrow$  (ctrWedgeIdx + i) mod numWedges
15    if wedgeIdx  $\in$  wedges then
16      wedge  $\leftarrow$  wedges[wedgeIdx]
17      if wedge.intersects(targetPoly) then
18        wedgeHits  $\leftarrow$  wedgeHits + 1
19        if target is a VEHICLE then
20          toRemove  $\leftarrow$  toRemove  $\cup$  wedgeIdx
21  if wedgeHits  $\geq$  minHit then
22    notOccluded  $\leftarrow$  notOccluded  $\cup$  target.trackToken
23  for key  $\in$  toRemove do
24    delete(wedges[key])
25 return notOccluded
```

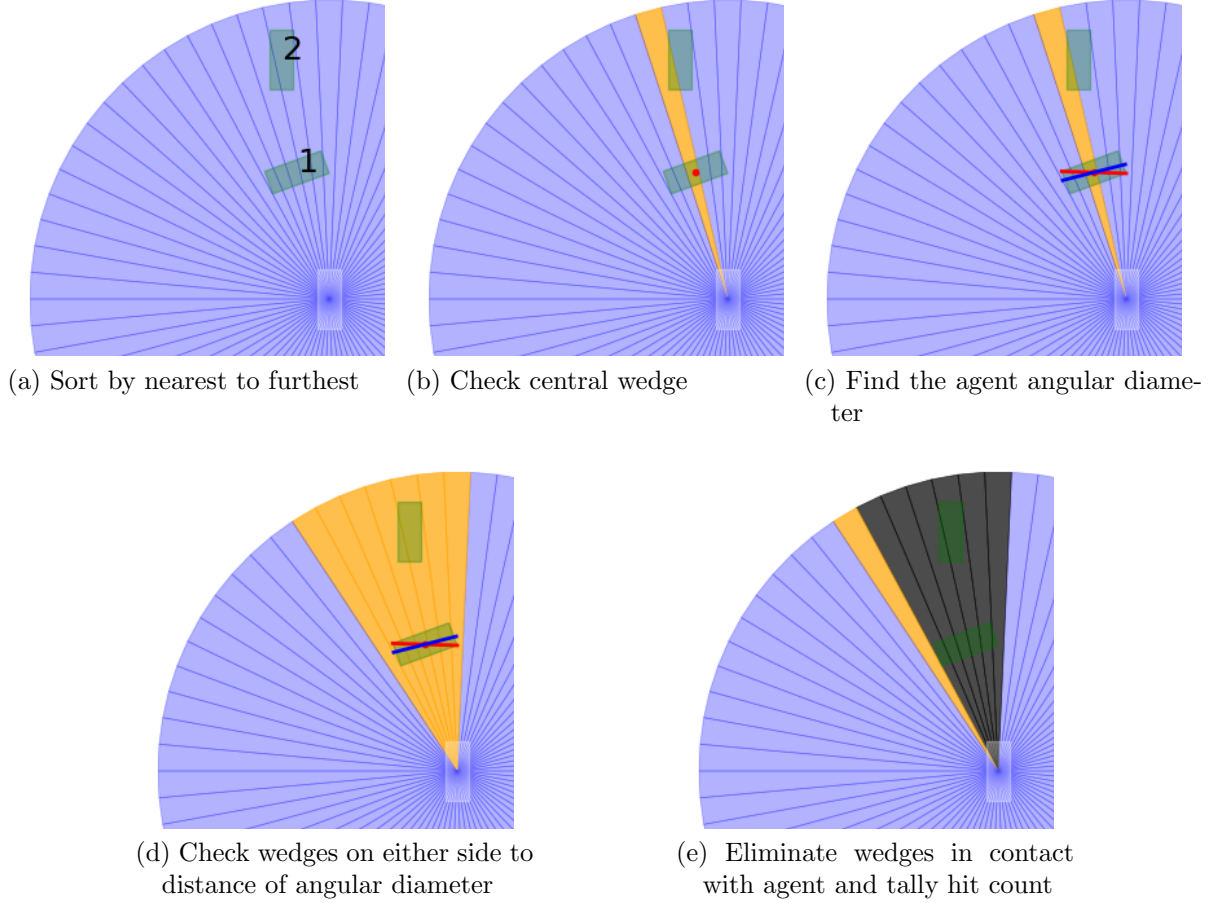


Figure 4.2: The steps taken to detect which objects are occluded. (a) The agents are processed nearest to furthest. (b) The wedge pointing to the centre of the current agent is found. (c) Once the angular diameter of the agent is found, (d) the number of wedges to check to either side of the wedge pointing to the centre of the agent is determined. We continue with the next step even if the initial wedge is no longer in our set. If enough wedges intersect the agent, it is determined to be visible. (e) Wedges that intersect our agent are removed from the overall set of wedges, meaning that objects in the resultant shadow will be occluded.

To model evidence accumulation and realistic reaction times, we implement the *notice system*, illustrated in Fig. 4.3. In order for an object to be visible to a planner at a given time-step in the current observation history buffer (OHB) of its trajectory, it must be

both visible (as determined by the aforementioned occlusion detection algorithm), and “noticed”. We implement the notice system by using two tunable values that we call the *uncloak window* (tuned to be 1.5s) and the *notice threshold* (tuned to be 1.0s). To be noticed, an object must satisfy two conditions. First, it must have been visible, or already noticed at the beginning of the uncloak window, which ends at the present time. This condition simulates the reaction time of the system from the first sight of the object to it being noticed and reacted to, which is equal to the uncloak window duration. Second, within the uncloak window, the object must be visible for at least the duration of the notice threshold (though not necessarily contiguously) in order to stop vehicles behind imperfect occlusions that only flicker into view for a single frame, from being detected. This simulates evidence accumulation and how vehicles only visible for a short period of time can be missed by humans, or classed as false positives by ADSs. The uncloak window must be greater than or equal to the notice threshold. In this way, it is possible for objects at older time-steps, which were not visible before, to become visible at later time-steps.

4.7 Scenario Generation

Our techniques for generating scenarios from the conflict taxonomy classes all rely on the following basic steps. We select a scenario from the dataset, insert a vehicle on a trajectory that will be in conflict with the ego vehicle, and then insert another vehicle in such a position that it occludes the conflict vehicle. The specific insertion strategies vary based on the type of OCC we are trying to generate. While our tools allow generators for any type of scenario to be created, our work has focused on creating a separate generator for each type of conflict vehicle position, including Sequential, Cross, Merge, and Diverge. Our OCCs are all generated on intersections, or areas where lanes cross, merge, or diverge from one another. While both the conflict vehicle and the occluder is inserted, care is taken to insert them into plausible locations, with plausible speeds and goals.

Guaranteeing realism is a serious challenge in this type of scenario generation, as the types of scenarios being generated exist in the long tails of the scenario distribution. Any comparison based on average driving measures is thus not likely to be relevant, as we are specifically looking at abnormal situations. To give the best chance of the scenarios being realistic, initial goals and speeds are set based on the goals and speeds of other vehicles in the same or upcoming lanes. Where no vehicles are available, speeds are set to respect the speed limits. Further, the vehicle driving behavior is simulated using tuPlan [12] which, being the gold medal winner in the nuPlan challenge, behaves closely to the vehicles in the recorded traffic.

t=6	1	2	3	4	5	6	7	8
Visible	✗	✗	✗	✓	✗	✓		
Noticed	✗	✗	✗	✗	✗	✗		
OHB	✗	✗	✗	✗	✗	✗		

↓

t=7	1	2	3	4	5	6	7	8
Visible	✗	✗	✗	✓	✗	✓	✓	
Noticed	✗	✗	✗	✗	✗	✗	✗	
OHB	✗	✗	✗	✗	✗	✗	✗	

↓

t=8	1	2	3	4	5	6	7	8
Visible	✗	✗	✗	✓	✗	✓	✓	✗
Noticed	✗	✗	✗	✓	✓	✓	✓	✓
OHB	✗	✗	✗	✓	✗	✓	✓	✗

Figure 4.3: The view of the present and past that the motion planner is given of this object. OHB refers to “Observation History Buffer”, provided to the planner. The shaded blue region visually displays the uncloak window. This example demonstrates the notice system with a 5-frame uncloak window and a 3-frame notice threshold. The object only becomes visible in the observation history at time-step 8, meeting both conditions of 3 frames of visibility and being visible at the start of the notice window. This shows that an object can be noticed while invisible (e.g., step 5 during time-step 8) and can be noticed on an invisible step (e.g., time-step 8). Previously unnoticed time-steps can become noticed once the system processes the appearance, revealing the trajectory for all steps that are both visible and noticed.

4.7.1 Cross Conflict Type Generation

The construction of the cross-type scenarios and the merge-type scenarios only differ in how the conflict point is selected. The cross scenario generator ensures that only lane connectors that cross the planned path of the ego vehicle are chosen as potential conflicts, while the merge generator only selects lane connectors that are merging with ego’s planned path (excluding connectors that are currently on red lights). The general steps of our cross scenario generator are outlined below, as well as in Fig. 4.4.

4.7.2 Cross and Merge Conflict Type Generation

1. We begin by loading in a real life scenario like Fig. 4.4a. These scenarios are initially chosen by selecting only those that involve ego traversing an intersection.
2. We optionally remove all vehicles that are not ego, shown in Fig. 4.4b. Keeping other vehicles in the scene can sometimes give us more occluded areas, allowing for more varied occlusion scenarios, while removing them increases the chances of generating exactly the type of OCC that this generator is designed for (cross/merge).
3. We select lane connectors that cross the lane connectors on ego’s immediate path, and identify the conflict point for each one as shown in Fig. 4.4c.
4. For each conflict point, we generate candidate conflict vehicles at multiple different speeds. The speeds are chosen via a handful of offsets which are applied to the speed of the ego vehicle. Each conflict vehicle is positioned on the conflicting lane connector such that at its chosen speed, it will reach the conflict point at the same time as ego as shown in Fig. 4.4d. In order to filter out poor spawn locations, the vehicles are simulated forwards for a short period of time to ensure that they do not collide with ego prematurely. If the newly inserted vehicle is already occluded by other vehicles, then the process is done, and we call such a scenario a “natural occlusion”. Otherwise, we move on to the next step.
5. To determine potential occlusion insertion locations, we create an FOV polygon from the centre of the ego vehicle to the left and rightmost edges of the conflict vehicle we want to obscure, as shown in Fig. 4.4e. Obstacles on the road, including road edges and other cars are subtracted from the FOV polygon in order to remove areas that could result in invalid spawn points.
6. To determine the candidate locations to spawn our occluder, we then take an intersection of the centre-lines of the lane connectors and the remaining FOV polygon as shown in Fig. 4.4f.
7. The resulting lines remaining from the intersection are discretized into points, which then make up our candidates as shown in Fig. 4.4g.
8. For each candidate point, we attempt to insert an occluder as shown in Fig. 4.4h. To avoid generating large amounts of scenarios with similar occluders, a minimum distance between accepted insertion points is enforced. Inserted vehicles are aligned with the direction of travel of their lane connector.

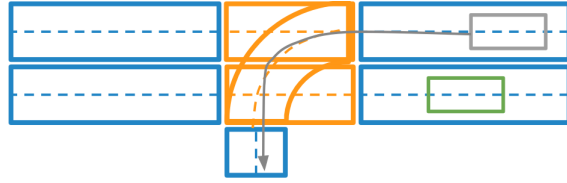
9. We create a polygon to catch other vehicles in the original scene. This polygon includes vehicles in the same lane connector, as well as outgoing lane connectors. All vehicles in this polygon (even if we removed them in the first step) are collected and sorted from nearest to furthest. Vehicles in adjacent lanes that are within a smaller range are also included, excepting those observing red lights. The occluder is assigned the speed of the nearest vehicle in this set. The goal is set the same way. If a valid vehicle does not exist, the speed limit is selected, and a random goal is given.
10. Having injected a potential occluder, we perform a check that the insertion actually does occlude our conflict vehicle. We repeat the premature collision check. If all checks pass, we submit the new scenario to the set of generated scenarios for testing.

4.7.3 Sequential Conflict Type Generation

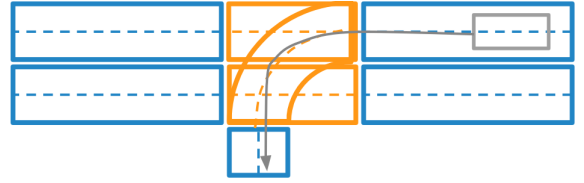
The construction of sequential scenarios follows a similar pattern. The key differences lie in the injection of the conflict vehicle. Unlike the cross scenarios, all sequential scenarios require the inserted conflict vehicle to be on the same lane as the ego vehicle, which severely reduces the number of possible scenarios that can be generated. These vehicles are inserted ahead of the ego vehicle, such that they are a fixed time gap ahead, and have their goals copied directly from ego. Their initial speed is set to 0, to simulate the conflict vehicle being unexpectedly stopped in the intersection, such as if they had to yield to a pedestrian or another vehicle. The process for inserting an occluding vehicle remains the same, except for the goal having to be selected so that the occluder takes a different route than ego, such that it can get out of the way and allow ego to have a chance to collide with the conflict vehicle.

4.7.4 Diverge Conflict Type Generation

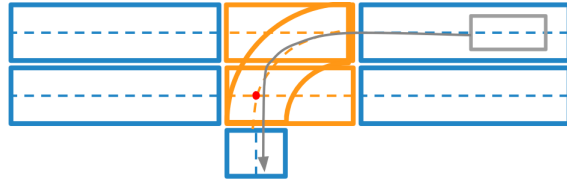
Diverge conflict scenarios share similarities with sequential scenarios. The roadways present in our dataset, combined with our particular choice of planner, make certain types of diverge conflicts infeasible to generate. Diverge-sequential type conflicts as depicted in the (diverge, sequential) cell of Table 3.1 requires the occluder to take the same path as the ego and bypass the diverging conflict vehicle which is still partially blocking the lane. While this is technically possible, our chosen planner tends to avoid moving laterally within the lane, instead preferring to slow down and wait for the obstruction to pass rather than bypassing it. Diverge-cross, diverge-merge and diverge-parallel type dynamic occlusions



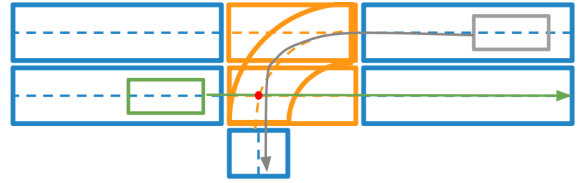
(a) Initial scenario



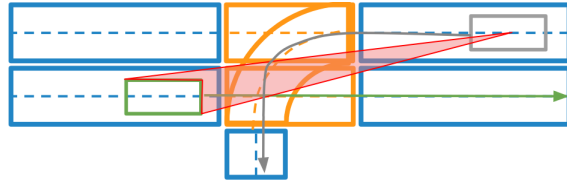
(b) Remove other vehicles (optional)



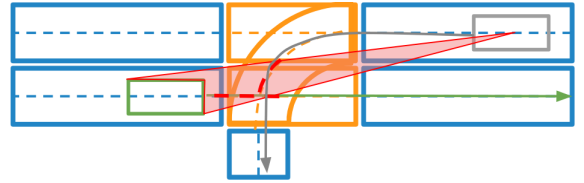
(c) Find conflict point (there may be multiple)



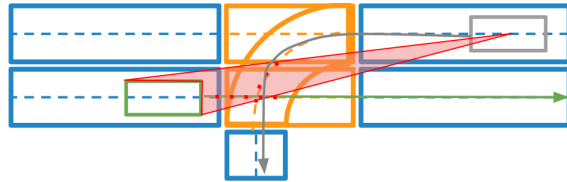
(d) Insert conflict agent (initial speed chosen according to other agents, position chosen to create a collision)



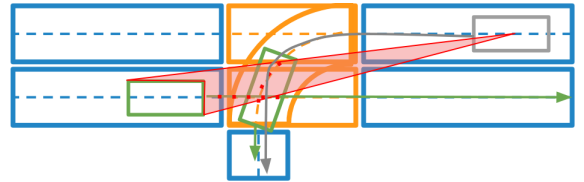
(e) Generate FOV polygon to conflict agent



(f) Find intersection of lane connector centre-lines with FOV polygon



(g) Discretize intersected centrelines into potential occluder spawn points



(h) Attempt to spawn new occluder at each point

Figure 4.4: The steps taken to generate an OCC using the cross conflict type generator. Ego is grey, other vehicles are green. Lanes are in blue, lane connectors are red. Planned paths are denoted by arrows.

require very particular lane geometries that do not exist within the nuPlan dataset in order to be generated. While it would be possible to generate such scenarios, it would require a different lane map.

As a result, our diverge type generator only attempts to generate diverge-diverge type dynamic occlusions. To that end, it makes several changes to the above algorithm used for sequential scenarios. It ensures that it only selects lanes that diverge into 3 different lane connectors in order to allow one lane for the conflict vehicle to diverge down, one for the occluder to diverge down (and not physically obstruct ego), and one for ego to attempt to drive down. Other differences include spawn locations being placed ahead of the conflict point rather than on it, such that the diverging vehicle leaves enough space for the occluder to bypass it when necessary while still obscuring the view of ego. The occluder’s goal is also set to diverge from both the conflict vehicle and ego where possible to aid the occluder in avoiding the conflict vehicle, as displayed in the (diverge, diverge) cell of Table 3.1.

Chapter 5

Experiments and Evaluation

Here, we explore the data and computational resource efficiency of our technique, as well as its ability to generate diverse OCCs. We determine the severity of the OCCs generated to show how vital such testing is, and observe the consequences of mistakes that even competition winning planners can make. Finally, we demonstrate the robustness of our method to faster reaction times to show that even a responsive system must take occlusions into account in order to guarantee the safety of everyone on the road.

5.1 Typology Coverage

We use a novel method to evaluate the variety that is achieved by our generator. Unlike previous work [24, 56] that categorizes and evaluates the type of scenarios they were capable of generating, we first create a typology of scenarios as shown in Table 3.1 and then evaluate how capable we are of generating the scenarios within that typology. Our generation techniques achieve significant coverage of the possible OCC types described in our typology. Table 5.1 displays the full coverage we were able to achieve across our generation testing, while Table 5.2, as a reference, displays the specific coverage of the configurations possible on a standard signalized two-lane cross intersection, that is, one with a single lane in each direction on each of the four intersection legs. Columns describe the relative motion of the conflicting vehicles, and rows describe the motion of the occluder relative to the conflicting vehicles. Cells marked with N/A denote configurations that are judged by the authors to not be geometrically feasible. We achieve a large, but not complete coverage over all types of OCCs thanks to our strategy of using four different generators, each one tuned to produce a single class. We achieve even better coverage when we limit

the possibilities to those that can be feasibly achieved on the reference intersection. Note that the nuPlan dataset contains complex multi-lane intersections, which allow covering additional OCC types compared to the reference intersection. We compare our coverage to the most closely related prior work [24] by mapping their generated scenarios according to their own typology on to our typology. In doing this, we find that we are capable of generating a larger portion of our new typology with our technique. This is shown in Table 5.1 with blue circles around the types that Kahn et al. [24] was able to generate.

Our weakness in generating scenarios where the occluder merges or is parallel is likely due to the small number of scenarios that occur on intersections with two parallel lanes in all directions, where a parallel or merging vehicle can obstruct ego’s view while still leaving a path open for the collision to occur between ego and the conflict vehicle. Our difficulty in generating OCCs where the conflict vehicle diverges is down to the geometric difficulty of creating a diverge conflict in the first place. Such a conflict relies on abnormal curving roadways before the intersection that are not commonly present in the dataset. As a result, these types of scenarios are difficult to generate for us.

Table 5.1: Generation coverage of the typology. Blue circles signify types that the most closely related prior work [24] was also capable of generating.

	Sequential	Merge	Diverge	Cross
Sequential	N/A			
Merge				
Diverge				
Cross				
Parallel				
Coverage	2/4	3/5	1/5	5/5

Table 5.2: Generation coverage of the typology on a signalized two-lane cross intersection.

	Sequential	Merge	Diverge	Cross
Sequential	N/A	✓	✗	✓
Merge	✗	✗	N/A	N/A
Diverge	✓	✓	✓	✓
Cross	✓	✓	✗	✓
Parallel	N/A	N/A	N/A	N/A
Coverage	2/3	3/4	1/3	3/3

5.2 Intersection Variety

The ability to generate collisions on a variety of intersections is a main advancement of this work. We thus introduce a second novel measure of variety in the number of intersections that we can create our scenarios on, relative to the number of scenarios generated. Compared to some previous works which focused on the automatic generation of OCCs on only one intersection [24], we show in Table 5.3 that we can generate OCCs on a wide variety of intersections contained in the nuPlan dataset [5]. Across our small scale testing runs, we pulled in many starting scenarios (called “original” in the second row of Table 5.3). Some of these scenarios exist on the same intersections, and so we filter the initial number down to show the number of unique intersections pulled in from the dataset for attempted generation, shown in the third row. From there, the generation process pares down the scenarios until we are left with a smaller, but still fairly diverse group of intersections, shown in the fourth row. The cross OCC generator has the least intersection diversity of the group, with only 10 unique intersections out of 73 OCCs. This is in mainly due to some scenarios being particularly fertile ground for generating cross type OCCs. Partly because some scenarios have a large amount of valid spawn positions and speeds for conflict vehicles that can result in an OCC, and partly because cross scenarios in particular allow more lanes to potentially host both the conflict and the occluding vehicle.

Table 5.3: Unique Intersection Generation

	Seq.	Merge	Diverge	Cross
# OCCs	22	12	7	73
# Original Scenarios	697	152	8789	152
# Unique Original Intersections	176	80	405	80
# Unique OCC Intersections	9	10	5	10

5.3 Generation Efficiency

Generation efficiency is the measure of how easy it is for a method to generate the desired type of scenario. We introduce two novel methods of measuring the generation efficiency. First is the data efficiency, which is described by the number of OCCs divided by the number of initial scenarios. It describes how many scenarios must be started with to generate a certain number of OCCs. The second method is to look at the computational efficiency. This is distinct from computational complexity, and simply examines how many OCCs are generated, divided by the number of total generated scenarios that must be tested using our dual filter method. A method that requires large amounts of input scenarios to generate a small amount of OCCs will be constrained by the cost of data collection. A method that generates large amounts of scenarios with a low probability of OCCs will need to spend more time and computational resources running and checking those scenarios. For these reasons, it is important to examine both measures of efficiency.

To abide by an upper limit of 1h of computation time per experiment, the number of initial scenarios input to each generator was varied. This limit covered both the time it took to generate the scenarios and the time it took to run them. The input scenarios themselves were all selected using the filters given in 4.3 with only the maximum number of each type varied.

Table 5.4 showcases the generation efficiencies for each of our 4 scenario generators, both when running in *full* and *reduced* modes, where reduced implies that all vehicles other than ego, the conflict vehicle and the occluder have been removed. Removing other vehicles in the scene has mixed results on the generation efficiency. It often results in more OCCs being generated from the same initial set of input scenarios, likely due to there being more free space on the road. Cross scenarios are the easiest to generate, largely due to the fast closing speed of vehicles and the large potential number of intersections, which gives rise to many scenarios that can be tested. Sequential scenarios are harder to generate scenarios for, but show a much higher rate of success per generated scenario.

Diverge scenarios are extremely difficult to generate, largely due to how accurately the three vehicles must be aligned to allow the occluder enough room to escape without also letting the conflict vehicles avoid the collision.

In comparison to the results shown in the table, prior work that only injected the occluder [24] managed to generate only 80 dynamic OCCs out of 105,914 scenarios, which were derived from 3,300 initial scenarios. This results in a computational efficiency of 0.0755% and a data efficiency of 2.424%. Our method, however, generates significantly more OCCs per scenario, even for the most challenging type of conflict (diverge), leading to better computational efficiency. Additionally, each of our generators achieves comparable or superior data efficiency compared to prior work.

Table 5.4: Generation Efficiency

R: Reduced F: Full	Initial Scenarios	Generated Scenarios	Ego-to-veh. Collisions	OCCs	OCCs Init'l Scens.	OCCs Gen'd Scens.
R Seq	697	52	26	22	3.16%	42.31%
F Seq	697	21	10	9	1.29%	42.86%
R Cross	152	1465	111	73	48.03%	4.98%
F Cross	152	634	42	20	13.16%	3.15%
R Merge	152	875	34	12	7.89%	1.37%
F Merge	152	475	20	10	6.58%	2.11%
R Diverge	8789	200	21	7	0.08%	3.50%
F Diverge	8789	159	25	9	0.10%	5.66%

5.4 Severity Analysis

The severity of an impact can be measured via the ΔV , or change in velocity, before and after the collision. The S0-3 severity scores outlined in [20] map onto the Abbreviated Injury Scale (AIS) [9] scores in Table 5.5 via Table 5.6. Prior work [24] used the same methodology, and when comparing using the same ΔV intervals, we obtained similar distributions of S0-3 severity scores in the generated OCCs for cross collisions, which were the most common type that were generated in that work. However, the ΔV intervals used in [24] were incorrectly chosen. In our work, we have chosen to use a different set of intervals from [52], defined in Table 5.7. It focuses on the more conservative of the ranges that were available to generate an upper bound on OCC severity. Unlike [24], we do not assume

the conflict vehicle that ego vehicle is colliding with is much heavier than ego to get our ΔV values. Instead, we assume a perfectly inelastic collision where both vehicles weigh roughly the same amount and calculate ΔV via Eq. (5.1), where the values are defined as follows: (m_A : Agent Mass, m_E : Ego Mass, v_A : Agent Speed, v_E : Ego Speed, θ : Heading Difference). Side collisions have lower bounds for severity because of the lower amount of crash structure between the occupants and the vehicles when compared to front or rear collisions.

$$[htb]\Delta V = \frac{m_A}{m_E + m_A} \sqrt{v_E^2 + v_A^2 - 2v_E v_A \cos(\theta)} \quad (5.1)$$

Collisions generated by the diverge and sequential collisions were the mildest because these collisions are all front to back as seen in Fig. 5.1, where the majority of the crash structure can provide protection. They also largely happen with both vehicles moving in the same direction, just at different speeds, which significantly brings down the ΔV from the beginning to the end of the crash. Both cross and merge generators generate a much wider variety of collision angles, which allows vehicles to strike in T-bone fashion, as well as achieve greater velocities at the moment of impact. The result is that these generators produce the most severe collisions, as seen in Fig. 5.2. The collisions produced by the cross generator are the ones most likely to resemble the left turn tag-on type collisions generated in [24] and [56].

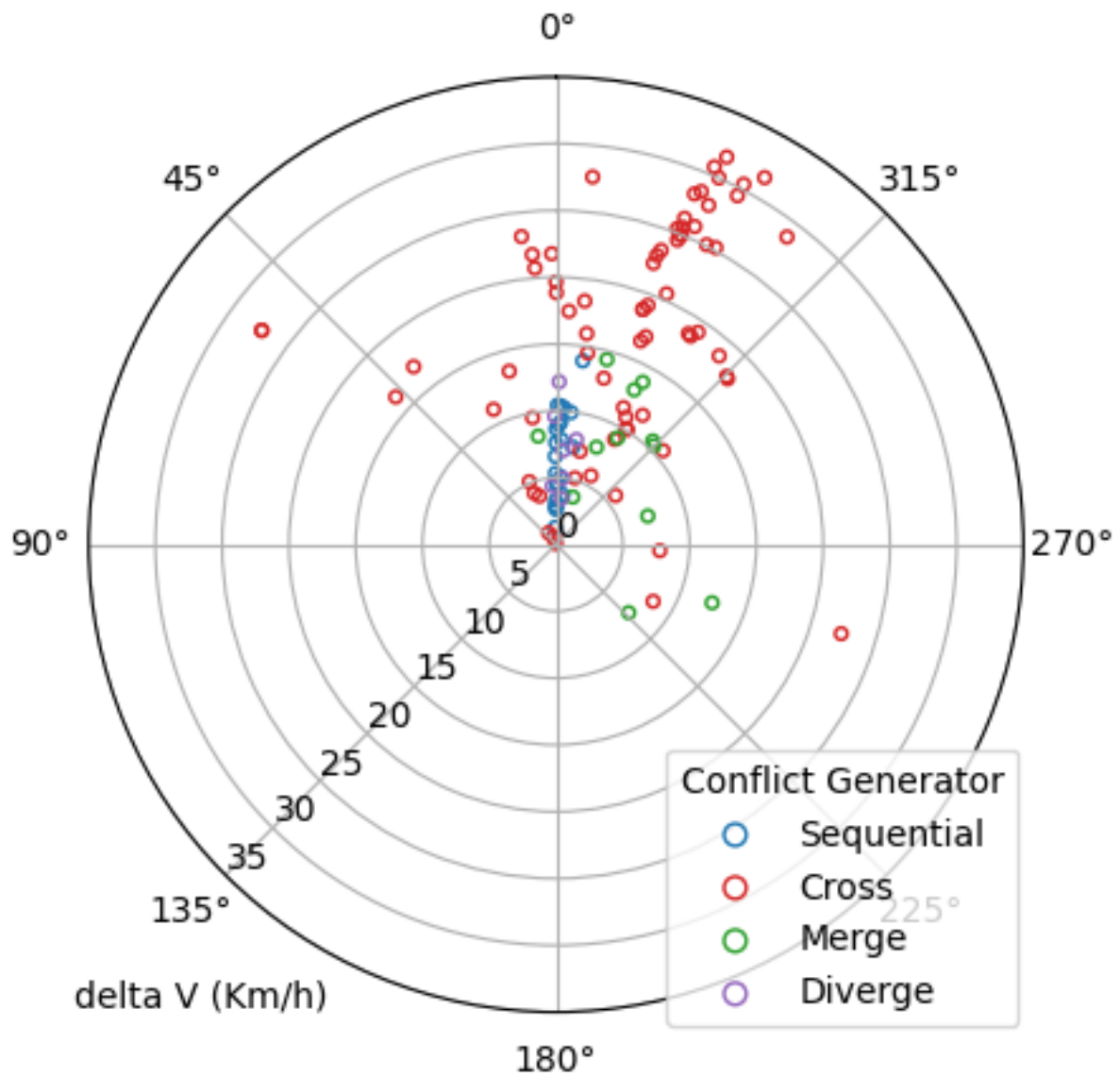


Figure 5.1: Radial plot of impact angles and ΔV for each generator. Impact angles are relative to ego's heading when ego collides with another vehicle in an OCC.

Table 5.5: The AIS injury scale categorizes severity from 1 (minor) to 6 (fatal) [9]

AIS Code	Injury	Example
1	Minor	Superficial Laceration
2	Moderate	Fractured Sternum
3	Serious	Open Fracture of Humerus
4	Severe	Perforated Trachea
5	Critical	Ruptured Liver with Tissue Loss
6	Maximum	Total Severance of Aorta

Table 5.6: Classification of Damage Based on Safety-Related Probability[52]

Category	Description
S0	No safety-related damage
S1	> 10% probability of AIS 1–6 (and not S2 or S3)
S2	> 10% probability of AIS 3–6 (and not S3)
S3	> 10% probability of AIS 5–6

Table 5.7: Collision Severity Categorization by Impact Side and ΔV Based on Conservative Intervals from [52]

Collision Side	ΔV Range km/h	Category
Front/Back	$s \leq 4$	0
	$4 < s \leq 20$	1
	$20 < s \leq 40$	2
	$40 < s$	3
Side	$s \leq 2$	0
	$2 < s \leq 8$	1
	$8 < s \leq 16$	2
	$16 < s$	3

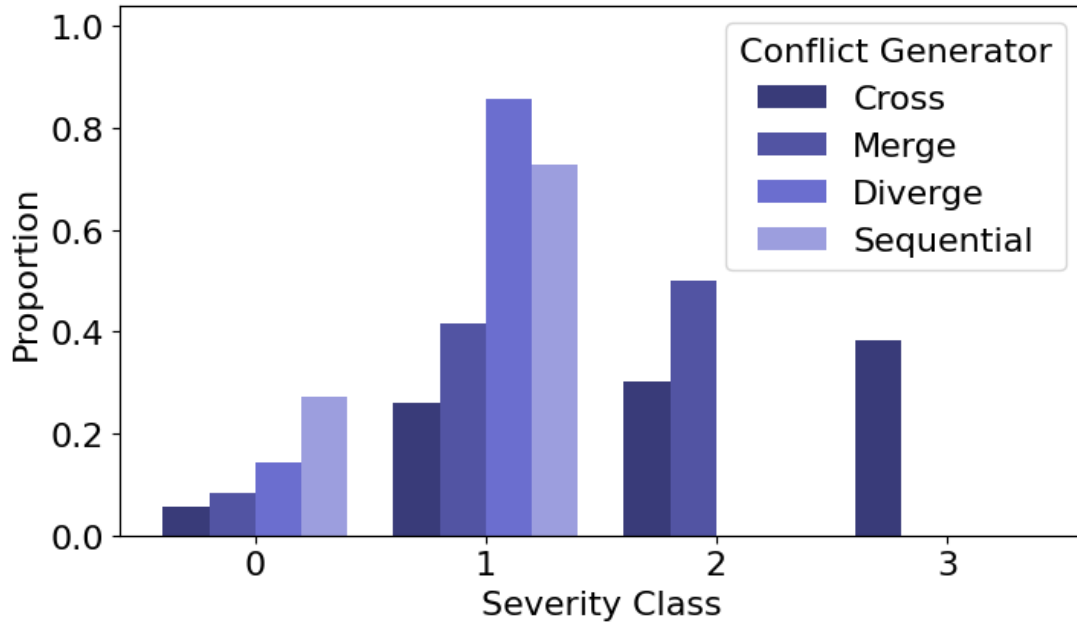


Figure 5.2: Severity class distributions of different types of OCC

5.5 Lone Occluder Insertion Ablation

Previous works [24] have operated by inserting only occluders into the scene. In order to compare this method of generation to ours, a pared-down version of our cross-conflict generator was created to see how many scenarios it could generate while selecting a potentially dangerous vehicle to ego, and then inserting an occluder that would block the view to ego. The number of OCCs generated could then be compared to the number of OCCs generated by our cross generator. Unfortunately, while this generator did produce candidate scenarios, it did not produce a single OCC, largely because having multiple vehicles arranged in such a way that they would collide if they continued onward is relatively rare without setting it up in each scene deliberately. Some scenes that are pulled from the dataset are also fairly sparse, without any vehicles close enough to be dangerous near the beginning of the scenario. Human drivers learn to avoid scenarios that risk resulting in a collision, and so few examples in the recorded traffic position themselves in a way that an inserted occluder can take advantage of. A more thorough search would be required to generate a few OCCs. It is therefore understandable why methods like those described in [24] required such a high number of generated scenarios to generate a similar number of OCCs as our own methods.

5.6 Reaction Time Ablations

Changes to reaction time have the potential to greatly increase or decrease the difficulty of generating OCCs. The chosen reaction time for all other experiments using our notice mechanism (Fig. 4.3) is an uncloak reaction time of 1.5 s and a notice threshold of 1 s. These values are chosen to be on the mid to lower end of human reaction time, since we are attempting to increase the likelihood of creating an OCC by constraining the amount of time a human has to react to the vehicle being obstructed by the dynamic occlusion. This can simulate a human panicking, being distracted, or tired. However, the ego vehicle, as well as any of the other vehicles on the road may be driven by an autonomous system, which may be capable of reacting more quickly. In order to test whether our generation methods are robust to more reactive agents, we lower both the uncloak reaction time and the notice threshold in our generated OCCs.

5.6.1 Ego Reaction Time

In Table 5.8, we gradually improve ego’s reaction time while maintaining the reaction time of all non-ego agents, while measuring the absolute number and percentage of previously generated OCCs that still result in a collision. At ego’s reaction time improves, simulating a world where ego is an ADS and everyone else is a human, we observe that some OCCs generated by each generator would not have occurred. Cross type OCCs are the most robust with respect to the improved reaction time and notice threshold.

Table 5.8: Percent of OCCs Still Occurring After Varying Ego Reaction Delay (Non-ego reactions set to (1.5, 1.0))

#OCCs %	(uncloak_reaction_time, notice_threshold)					
	(1.5 s, 1.0 s)		(1.0 s, 0.67 s)		(0.5 s, 0.33 s)	
Reduced Sequential	22	100%	12	54.55%	7	31.82%
Reduced Cross	73	100%	59	80.82%	47	64.38%
Reduced Merge	12	100%	9	75.00%	7	58.33%
Reduced Diverge	7	100%	2	28.57%	1	14.29%

5.6.2 All Agent Reaction Time

In Table 5.9, we start with the ego’s reaction time being lowered to the final rightmost column in Table 5.8, and gradually improve all other agent’s reaction times. This simulates a situation where all vehicles are driven by ADS, or by very attentive drivers. We see a similar result, with fewer OCCs still occurring in most categories. Sequential and diverge OCCs are largely unchanged due to the fact that the conflict vehicle starts stationary in all sequential and diverge scenarios ahead of both other vehicles. This means that the vehicles have little agency in the collision. In the final column, we actually observe an increase in the number of OCCs at the fastest reaction time. This can result if the extra information being visible earlier resulted in a slower acceleration of the initially stopped vehicle. In the cross and merge type scenarios, we see the same type of dropoff as observed in the previous table.

Table 5.9: Percent of OCCs Still Occurring After Varying Agent Reaction Delay (ego reactions set to (0.5, 0.33))

#OCCs %	(uncloak_reaction_time, notice_threshold)					
	(1.5 s, 1.0 s)		(1.0 s, 0.67 s)		(0.5 s, 0.33 s)	
Reduced Sequential	7	100%	7	100%	8	114.3%
Reduced Cross	47	100%	28	59.57%	22	46.81%
Reduced Merge	7	100%	3	42.86%	1	14.28%
Reduced Diverge	1	100%	1	100%	1	100%

Chapter 6

Threats to Validity and Future Work

In this chapter, we acknowledge a number of threats to the validity of our work related to our particular implementation. We also mention some as-yet unexplored extensions.

6.1 Threats to Validity

As seen in the second part of our reaction time ablations, in rare occasions some scenarios generated as OCCs may be the result of a planner reacting unpredictably to the on-road conditions rather than anything to do with the occlusion. Another limitation is the lack of direct traffic light data in the nuPlan dataset. The nuPlan simulator infers traffic light status on a per-lane basis based on recorded agent movement, which results in empty intersections always being green, and any lane with only a single stopped agent in the scene evaluating to red, even if the ground truth is different, which can result in spawn points that are less realistic than normal. Our diverge generator is perhaps the least general of our generators. Unlike our other generators, which have the possibility of generating any type of occluder motion within its conflict type, our diverge scenarios are limited to generating diverge-diverge scenarios such as those displayed in the (diverge, diverge) cell of Table 3.1. This was done due to a number of previously explained reasons that made generation of the other patterns difficult on this set of roadways, though it is entirely possible that a more thorough search of intersections would have yielded one where more of these patterns could have been created, thus increasing our methods coverage of scenario types within our typology.

6.2 Future Work

In the creation of this work, we identified several paths that we had not fully explored that may merit future exploration. We did not attempt to generate dynamic occlusions where the occluded agent is a pedestrian or cyclist, in part because pedestrians and all other non vehicle tracks are not controlled in a reactive manner in nuPlan, but instead simply part of recorded playback. This can result in pedestrians and cyclists making poor decisions when crossing the road and walking into stopped vehicles. We did not explore varying the size or type of the inserted vehicles, due to the already substantial combinatorial explosion of potential OCCs that would be generated. We also did not explore slight perturbations in the inserted positions of both the conflict vehicle, and the occluder despite the fact that it would make some OCCs possible where they had not been before for the same reason. Multiple different planners are available to control the ego and other agents in our experiments. Experimenting with what kind of OCCs can be generated with different planners and how choosing different planners for different vehicles would influence the results should be explored in future work. In small scale experiments, it was found, however, that generating OCCs for significantly weaker planners was less reliable, since weak planners would often move unpredictably and create OCCs simply because of poor decisions in the occluded regime that had nothing to do with the occlusion itself.

Chapter 7

Conclusion

In this work, we have introduced a method to generate dynamic occlusion scenarios crucial for safety validation in autonomous driving systems. Our approach demonstrates effective scenario generation across diverse road geometries, addressing the scarcity of critical occlusion scenarios in existing datasets. We introduce a typology of occlusion scenarios and achieve significant coverage in our ability to generate the various types of critical occlusions. Our severity analysis of the generated scenarios shows that they can be life-threatening, and are worth testing against to ensure ADSs can handle them. By leveraging our custom simulation modifications and scenario generation techniques, we successfully created a variety of occlusion-induced collision scenarios, enhancing the robustness of validation efforts. This method broadens the scope of safety testing and contributes to advancing the safety of autonomous driving systems. We hope that this work can prompt more explicit occlusion testing to be made part of the testing procedures of the ADS industry.

References

- [1] Arpit S Agarkar, R Gandhiraj, Manoj Kumar Panda, and Saksham Srivastava. Simulation based closed loop scenario fuzzing for safety critical ADAS applications. In *2023 IEEE 20th India Council International Conference (INDICON)*, pages 759–764, December 2023. ISBN 2325-9418. doi: 10.1109/INDICON59947.2023.10440801.
- [2] Matthias Althoff, Markus Koschi, and Stefanie Manzing. Commonroad: Composable benchmarks for motion planning on roads. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 719–726, 2017. doi: 10.1109/IVS.2017.7995802.
- [3] Sebastian Brechtel, Tobias Gindele, and Rüdiger Dillmann. Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps. In *17th international IEEE conference on intelligent transportation systems (ITSC)*, pages 392–399. IEEE, 2014.
- [4] Nancy L. Broen and Dean P. Chiang. Braking response times for 100 drivers in the avoidance of an unexpected obstacle as measured in a driving simulator. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 40(18):900–904, 1996. doi: 10.1177/154193129604001807.
- [5] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. NuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles, February 2022. arXiv:2106.11810.
- [6] Jinkang Cai, Weiwen Deng, Haoran Guang, Ying Wang, Jiangkun Li, and Juan Ding. A survey on data-driven scenario generation for automated vehicle testing. *Machines*, 10(11), 2022. ISSN 2075-1702. doi: 10.3390/machines10111101. URL <https://www.mdpi.com/2075-1702/10/11/1101>.
- [7] Jinkang Cai, Shichun Yang, and Haoran Guang. A review on scenario generation for testing autonomous vehicles. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 3371–3376. IEEE, 2024.

- [8] Filippos Christianos, Peter Karkus, Boris Ivanovic, Stefano V Albrecht, and Marco Pavone. Planning with occluded traffic agents using bi-level variational occlusion models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5558–5565. IEEE, 2023.
- [9] Wayne S Copes, Howard R Champion, William J Sacco, Mary M Lawnick, Donald S Gann, Thomas Gennarelli, Ellen MacKenzie, and Steven Schwaitzberg. Progress in characterizing anatomic injury. *Journal of Trauma and Acute Care Surgery*, 30(10): 1200–1207, 1990.
- [10] Krzysztof Czarnecki. Operational world model ontology for automated driving systems - Part 1: Road structure. Technical report, Waterloo Intelligent Systems Engineering (WISE) Lab, University of Waterloo, 2018.
- [11] Krzysztof Czarnecki. Operational world model ontology for automated driving systems - Part 2: Road users, animals, other obstacles, and environmental conditions. Technical report, Waterloo Intelligent Systems Engineering (WISE) Lab, University of Waterloo, 2018.
- [12] Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting with misconceptions about learning-based vehicle motion planning, 2023. arXiv: 2306.07962.
- [13] Joost C F de Winter, Mehdi Saffarian, and John W Senders. The effect of an occlusion-induced delay on braking behavior in critical situations: A driving simulator study. *Human Factors*, 65(7):1336–1344, 2023. doi: 10.1177/00187208221101301.
- [14] Wenhao Ding, Chejian Xu, Mansur Arief, Haohong Lin, Bo Li, and Ding Zhao. A survey on safety-critical driving scenario generation—a methodological perspective. *IEEE Transactions on Intelligent Transportation Systems*, 24(7):6971–6988, 2023. doi: 10.1109/TITS.2023.3259322.
- [15] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.
- [16] Paweł Drożdżel, Sławomir Tarkowski, Iwona Rybicka, and Rafał Wrona. Drivers’ reaction time research in the conditions in the real traffic. *Open Engineering*, 10(1): 35–47, 2020. doi: doi:10.1515/eng-2020-0004.

- [17] Eli-Henry Dykhne, Shea Anthony Cardozo, James Riddell, Sean Sedwards, and Krzysztof Czarnecki. Occ-gen: Automated generation of dynamic occlusion-caused collisions. *Under submission*, 2024.
- [18] Johan Engström, Shu-Yuan Liu, Azadeh DinparastDjadid, and Camelia Simoiu. Modeling road user response timing in naturalistic traffic conflicts: a surprise-based framework. *Accident Analysis & Prevention*, 198:107460, 2024.
- [19] Marcel Hallgarten, Martin Stoll, and Andreas Zell. From prediction to planning with goal conditioned lane graph traversals, 2023. arXiv: 2302.07753.
- [20] ISO Central Secretary. Road vehicles — functional safety — part 3: Concept phase. Standard ISO TR 26262-3:2018, International Organization for Standardization, Geneva, CH, 2018.
- [21] J. Ge, J. Zhang, C. Chang, Y. Zhang, D. Yao, Y. Tian, and L. Li. Dynamic Testing for Autonomous Vehicles Using Random Quasi Monte Carlo. *IEEE Transactions on Intelligent Vehicles*, 9(3):4480–4492, March 2024. ISSN 2379-8904. doi: 10.1109/TIV.2024.3358329.
- [22] J. Ge, P. Wang, C. Chang, Y. Zhang, D. Yao, and L. Li. Life-long Learning and Testing for Automated Vehicles via Adaptive Scenario Sampling as A Continuous Optimization Process. *IEEE Transactions on Intelligent Vehicles*, pages 1–14, 2024. ISSN 2379-8904. doi: 10.1109/TIV.2024.3401186.
- [23] J. Ransiek, B. Schütt, A. Hof, and E. Sax. Generation of Adversarial Trajectories using Reinforcement Learning to Test Motion Planning Algorithms. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2819–2826, September 2023. ISBN 2153-0017. doi: 10.1109/ITSC57777.2023.10422130. Journal Abbreviation: 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC).
- [24] Maximilian Kahn, Atrisha Sarkar, and Krzysztof Czarnecki. I know you can’t see me: Dynamic occlusion-aware safety validation of strategic planners for autonomous vehicles using hypergames. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 11202–11208, May 2022. doi: 10.1109/ICRA46639.2022.9812041.
- [25] Arne Kesting, Martin Treiber, and Dirk Helbing. General lane-changing model mobil for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007.

- [26] Markus Koschi and Matthias Althoff. Set-based prediction of traffic participants considering occlusions and traffic rules. *IEEE Transactions on Intelligent Vehicles*, 6(2): 249–265, 2020.
- [27] Kristofer Kusano and Trent Victor. Methodology for determining maximum injury potential for automated driving system evaluation. *Traffic Injury Prevention*, 23(sup1): S224–S227, October 2022. doi: 10.1080/15389588.2022.2125231.
- [28] Kristofer D Kusano, John M Scanlon, Mattias Brännström, Johan Engström, and Trent Victor. Framework for a conflict typology including contributing factors for use in ads safety evaluation. In *27th International Technical Conference on the Enhanced Safety of Vehicles (ESV) National Highway Traffic Safety Administration*, 2023. 23-0328.
- [29] L. Feng, Q. Li, Z. Peng, S. Tan, and B. Zhou. TrafficGen: Learning to Generate Diverse and Realistic Traffic Scenarios. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3567–3575, June 2023. doi: 10.1109/ICRA48891.2023.10160296. Journal Abbreviation: 2023 IEEE International Conference on Robotics and Automation (ICRA).
- [30] Aliaksei Lareshyn and András Várhelyi. The Swedish traffic conflict technique: Observer’s manual. Technical report, Lund University, 2018.
- [31] Neil D. Lerner. Brake perception-reaction times of older and younger drivers. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 37(2):206–210, 1993. doi: 10.1177/154193129303700211.
- [32] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *2018 21st international conference on intelligent transportation systems (ITSC)*, pages 2575–2582. IEEE, 2018.
- [33] Bharath Bhaskar Mahadikar, Nishant Rajesh, Kevin Tom Kurian, Erjen Lefebber, Jeroen Ploeg, Nathan van de Wouw, and Mohsen Alirezai. Formulating a dissimilarity metric for comparison of driving scenarios for automated driving systems. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 1091–1098. IEEE, 2024.
- [34] Gustav Markkula. Modeling driver control behavior in both routine and near-accident driving. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 58, pages 879–883. SAGE Publications Sage CA: Los Angeles, CA, 2014.

- [35] Timothy L. McMurtry, Joseph M. Cormier, Tom Daniel, John M. Scanlon, and Jeff R. Crandall. An omni-directional model of injury risk in planar crashes with application for autonomous vehicles. *Traffic Injury Prevention*, 22(sup1):S122–S127, October 2021. doi: 10.1080/15389588.2021.1955108.
- [36] Ministry of Transportation. Geometric design standards for Ontario highways, 1985. Survey and Design Office, Ontario, Canada.
- [37] Korbinian Moller, Rainer Trauth, and Johannes Betz. Overcoming blind spots: Occlusion considerations for improved autonomous driving safety. *arXiv preprint arXiv:2402.01507*, 2024.
- [38] Nico Montali, John Lambert, Paul Mouglin, Alex Kuefler, Nick Rhinehart, Michelle Li, Cole Gulino, Tristan Emrich, Zoey Yang, Shimon Whiteson, Brandyn White, and Dragomir Anguelov. The Waymo open sim agents challenge, 2023. arXiv: 2305.12032.
- [39] H. Muslim, S. Endo, H. Imanaga, S. Kitajima, N. Uchida, E. Kitahara, K. Ozawa, H. Sato, and H. Nakamura. Cut-out scenario generation with reasonability foreseeable parameter range from real highway dataset for autonomous vehicle assessment. *IEEE Access*, 11:45349–45363, 2023. doi: 10.1109/ACCESS.2023.3268703.
- [40] N. Rajesh, E. Van Hassel, and M. Alirezaei. Unknown-Unsafe Scenario Generation for Verification and Validation of Automated Vehicles. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1759–1764, September 2023. ISBN 2153-0017. doi: 10.1109/ITSC57777.2023.10421804. Journal Abbreviation: 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC).
- [41] Patiphon Narksri, Hatem Darweesh, Eijiro Takeuchi, Yoshiki Ninomiya, and Kazuya Takeda. Occlusion-aware motion planning with visibility maximization via active lateral position adjustment. *IEEE Access*, 10:57759–57782, 2022.
- [42] National Highway Traffic Safety Administration (NHTSA). National motor vehicle crash causation survey: Report to congress, 2008.
- [43] Emma J. Nilsson, Mikael Ljung Aust, Johan Engström, Bo Svanberg, and Per Lindén. Effects of cognitive load on response time in an unexpected lead vehicle braking scenario and the detection response task (drt). *Transportation Research Part F: Traffic Psychology and Behaviour*, 59:463–474, 2018. doi: <https://doi.org/10.1016/j.trf.2018.09.026>.

- [44] Piotr F Orzechowski, Annika Meyer, and Martin Lauer. Tackling occlusions & limited sensor range with set-based safety verification. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1729–1736. IEEE, 2018.
- [45] Charles Packer, Nicholas Rhinehart, Rowan Thomas McAllister, Matthew A Wright, Xin Wang, Jeff He, Sergey Levine, and Joseph E Gonzalez. Is anyone there? learning a planner contingent on perceptual uncertainty. In *Conference on Robot Learning*, pages 1607–1617. PMLR, 2023.
- [46] Elisabeth J Ploran, Steven M Nelson, Katerina Velanova, David I Donaldson, Steven E Petersen, and Mark E Wheeler. Evidence accumulation and the moment of recognition: dissociating perceptual recognition processes using fmri. *Journal of Neuroscience*, 27(44):11912–11924, 2007.
- [47] R. Kensbock, M. Nezami, and G. Schildbach. Scenario-Based Decision-Making, Planning and Control for Interaction-Aware Autonomous Driving on Highways. In *2023 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–6, June 2023. ISBN 2642-7214. doi: 10.1109/IV55152.2023.10186546. Journal Abbreviation: 2023 IEEE Intelligent Vehicles Symposium (IV).
- [48] Davis Rempe, Jonah Philion, Leonidas J. Guibas, Sanja Fidler, and Or Litany. Generating useful accident-prone driving scenarios via a learned traffic prior, 2022. arXiv:2112.05077.
- [49] Stefan Riedmaier, Thomas Ponn, Dieter Ludwig, Bernhard Schick, and Frank Diermeyer. Survey on scenario-based safety assessment of automated vehicles. *IEEE access*, 8:87456–87477, 2020.
- [50] Ricardo Roriz, Jorge Cabral, and Tiago Gomes. Automotive lidar technology: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):6282–6297, 2021.
- [51] S. Jing, Y. Zhao, X. Zhao, F. Hui, and A. J. Khattak. An Efficient High-Risk Lane-Changing Scenario Edge Cases Generation Method for Autonomous Vehicle Safety Testing. *IEEE Transactions on Intelligent Vehicles*, pages 1–13, 2024. ISSN 2379-8904. doi: 10.1109/TIV.2024.3414828.
- [52] *Considerations for ISO 26262 ASIL Hazard Classification*. SAE, 2018. J2980_201804.
- [53] José Manuel Gaspar Sánchez, Truls Nyberg, Christian Pek, Jana Tumova, and Martin Törngren. Foresee the unseen: Sequential reasoning about hidden obstacles for safe

- driving. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 255–264. IEEE, 2022.
- [54] Oliver Scheel, Luca Bergamini, Maciej Wołczyk, Błażej Osiński, and Peter Ondruska. Urban driver: Learning to drive from real-world demonstrations using policy gradients, 2021. arXiv: 2109.13333.
 - [55] Philip Schörner, Lars Töttel, Jens Doll, and J Marius Zöllner. Predictive trajectory planning in situations with hidden road users using partially observable markov decision processes. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 2299–2306. IEEE, 2019.
 - [56] Hao Shao, Letian Wang, Ruobing Chen, Steven L. Waslander, Hongsheng Li, and Yu Liu. ReasonNet: End-to-end driving with temporal and global reasoning, 2023. arXiv: 2305.10507.
 - [57] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000.
 - [58] W. Jiang, J. Chen, Y. Ma, L. Zhang, and N. Zhong. A Generation Method of Unknown Unsafe Scenarios for Autonomous Vehicles. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1920–1926, September 2023. ISBN 2153-0017. doi: 10.1109/ITSC57777.2023.10422389. Journal Abbreviation: 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC).
 - [59] Denggui Wang, Weiping Fu, Jincan Zhou, and Qingyuan Song. Occlusion-aware motion planning for autonomous driving. *IEEE Access*, 11:42809–42823, 2023.
 - [60] Jingkan Wang, Ava Pun, James Tu, Sivabalan Manivasagam, Abbas Sadat, Sergio Casas, Mengye Ren, and Raquel Urtasun. AdvSim: Generating safety-critical scenarios for self-driving vehicles. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9904–9913, June 2021. doi: 10.1109/CVPR46437.2021.00978.
 - [61] X. Gu, S. Wu, B. Li, H. Wang, X. Jiao, Z. Wang, W. Lu, W. Yu, L. Chen, J. Hu, and S. Zhao. Copula-Based Interactive Behavior Modeling for Naturalistic Test Scenarios Generation. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 5940–5947, September 2023. ISBN 2153-0017. doi: 10.1109/ITSC57777.2023.10421919. Journal Abbreviation: 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC).

- [62] X. Li, E. Liu, T. Shen, J. Huang, and F. -Y. Wang. ChatGPT-Based Scenario Engineer: A New Framework on Scenario Generation for Trajectory Prediction. *IEEE Transactions on Intelligent Vehicles*, 9(3):4422–4431, March 2024. ISSN 2379-8904. doi: 10.1109/TIV.2024.3363232.
- [63] Chejian Xu, Ding Zhao, Alberto Sangiovanni-Vincentelli, and Bo Li. Diffscene: Diffusion-based safety-critical scenario generation for autonomous vehicles. In *The Second Workshop on New Frontiers in Adversarial Machine Learning*, 2023.
- [64] Y. Li, S. Wu, and H. Wang. Adaptive Mining of Failure Scenarios for Autonomous Driving Systems Based on Multi-population Genetic Algorithm. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 2458–2464, June 2024. ISBN 2642-7214. doi: 10.1109/IV55156.2024.10588541. Journal Abbreviation: 2024 IEEE Intelligent Vehicles Symposium (IV).
- [65] Y. Yang, K. Kujanpää, I. A. Babadi, J. Pajarinen, and A. Ilin. Suicidal Pedestrian: Generation of Safety-Critical Scenarios for Autonomous Vehicles. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1983–1988, September 2023. ISBN 2153-0017. doi: 10.1109/ITSC57777.2023.10422034. Journal Abbreviation: 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC).
- [66] Y. Zhu, J. Wang, X. Guo, F. Meng, and T. Liu. Functional Testing Scenario Library Generation Framework for Connected and Automated Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 24(9):9712–9724, September 2023. ISSN 1558-0016. doi: 10.1109/TITS.2023.3266639.
- [67] Ming-Yuan Yu, Ram Vasudevan, and Matthew Johnson-Roberson. Occlusion-aware risk assessment for autonomous driving in urban environments. *IEEE Robotics and Automation Letters*, 4(2):2235–2241, 2019.
- [68] Z. He, J. Zhang, D. Yao, Y. Zhang, and H. Pei. Adversarial Generation of Safety-Critical Lane-Change Scenarios for Autonomous Vehicles. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 6096–6101, September 2023. ISBN 2153-0017. doi: 10.1109/ITSC57777.2023.10422684. Journal Abbreviation: 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC).
- [69] Z. Wang, X. Li, D. Wei, L. Wang, and Y. Huang. Efficient Generation of Safety-Critical Scenarios Combining Dynamic and Static Scenario Parameters. *IEEE Transactions*

on *Intelligent Vehicles*, pages 1–16, 2024. ISSN 2379-8904. doi: 10.1109/TIV.2024.3402221.

- [70] Z. Wei, H. Huang, G. Zhang, R. Zhou, X. Luo, S. Li, and H. Zhou. Interactive Critical Scenario Generation for Autonomous Vehicles Testing Based on In-depth Crash Data Using Reinforcement Learning. *IEEE Transactions on Intelligent Vehicles*, pages 1–12, 2024. ISSN 2379-8904. doi: 10.1109/TIV.2024.3415961.
- [71] Zixu Zhang and Jaime Fisac. Safe occlusion-aware autonomous driving via game-theoretic active perception. In *Robotics: Science and Systems XVII*, RSS2021. Robotics: Science and Systems Foundation, July 2021. doi: 10.15607/rss.2021.xvii.066.
- [72] Zixu Zhang and Jaime F Fisac. Safe occlusion-aware autonomous driving via game-theoretic active perception. *arXiv preprint arXiv:2105.08169*, 2021.
- [73] Ming Zhou, Jun Luo, Julian Villella, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, Montgomery Alban, Iman Fadakar, Zheng Chen, Aurora Chongxi Huang, Ying Wen, Kimia Hassanzadeh, Daniel Graves, Dong Chen, Zhengbang Zhu, Nhat Nguyen, Mohamed Elsayed, Kun Shao, Sanjeevan Ahilan, Baokuan Zhang, Jiannan Wu, Zhengang Fu, Kasra Rezaee, Peyman Yadmellat, Mohsen Rohani, Nicolas Perez Nieves, Yihan Ni, Seyedershad Banijamali, Alexander Cowen Rivers, Zheng Tian, Daniel Palenicek, Haitham bou Ammar, Hongbo Zhang, Wulong Liu, Jianye Hao, and Jun Wang. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving, 11 2020. URL <https://arxiv.org/abs/2010.09776>.