

UC Santa Cruz

UC Santa Cruz Electronic Theses and Dissertations

Title

Controllable and Interpretable Generation of Realistic Autonomous Vehicle Test Scenarios.

Permalink

<https://escholarship.org/uc/item/1qt3h5fv>

ISBN

9798288887048

Author

Paranjape, Ishaan

Publication Date

2025-06-12

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
SANTA CRUZ

**CONTROLLABLE AND INTERPRETABLE GENERATION OF
REALISTIC AUTONOMOUS VEHICLE TEST SCENARIOS.**

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTATIONAL MEDIA

by

Ishaan Paranjape

June 2025

The Dissertation of Ishaan Paranjape
is approved:

Prof. Jim Whitehead, Chair

Asst. Teaching Prof. Markus Eger

Assoc. Prof. Adam M. Smith

Peter Biehl
Vice Provost and Dean of Graduate Studies

Copyright © by
Ishaan Paranjape
2025

Table of Contents

List of Figures	v
List of Tables	viii
Abstract	ix
1 Introduction	1
1.1 Research Questions	4
2 A meta-model for Scenario-based Testing of Autonomous Vehicles	7
2.1 Perspectives on Autonomy	10
2.1.1 Behavioral Autonomy	11
2.1.2 Technical Autonomy	11
2.1.3 Intra-component Impact: Relations between different autonomy perspectives	12
2.1.4 Inter-component Impact: Expectations from AV Test Scenarios	12
2.2 Perspectives on AV Test Scenarios	13
2.2.1 Conceptual Perspectives	13
2.2.2 Scenario Dataset Perspectives	16
2.2.3 Scenario Generation Perspectives	17
2.2.4 Scenario Evaluation Metrics	19
2.2.5 Simulation Tool Perspectives	20
2.2.6 Intra-component Impact	22
2.2.7 Inter-component Impact	23
2.3 Conclusion	24
3 Cruzway RoadGen: Designer-Encoded Realistic Road Generation	25
3.1 Introduction	25
3.2 Cruzway	27

3.2.1	High Definition Road Generation: Enhanced Netgenerate .	29
3.2.2	Realistic Road Generation: Sumo2Unreal	30
3.2.3	Blender OpenDRIVE Importer	31
3.3	Evaluation and Results	32
3.4	Limitations	38
3.5	Conclusion	38
4	CruzwayQA: Benchmarking Scenario Reasoning Capabilities in LLMs	39
4.1	Introduction	39
4.2	Preliminaries	44
4.3	Method	45
4.3.1	Scenario Ontology	46
4.4	Experimental Setup	87
4.4.1	Controllability and Diversity	89
4.4.2	LLM performance	90
4.5	Results	90
4.5.1	Alignment	92
4.5.2	Controllability and Diversity	94
4.5.3	LLM performance	114
4.6	Conclusion	114
5	Cruzway Scenario Generator	117
5.1	Introduction	117
5.2	Method	125
5.2.1	Dataset and Preprocessing	126
5.2.2	Prompting Using Model based Task Planning	127
5.2.3	Response Evaluations using LLM as a Judge Approach . .	128
5.2.4	Diverse Scenario Generation	129
5.3	Evaluation	129
5.4	Conclusion	133
	Bibliography	135

List of Figures

1.1	The promise of LLM-based test scenario generation	4
1.2	A five way intersection generated using Cruzway RoadGen.	6
2.1	A high level overview of the meta-model.	9
2.2	Clarifying perspectives helps us identify aspects to formalize for scenario generation using standardized forms.	9
2.3	Differing perspectives can affect the focal aspects of the two com- ponents of the meta-model.	10
3.1	The system architecture of CruzWay RoadGen	28
3.2	A road intersection generated by blenderOpenDRIVEImporter . .	31
3.3	A road intersection generated by blenderOpenDRIVEImporter im- ported into the CARLA simulator	32
3.4	A four way intersection generated using IntGen and rendered on Unreal Engine using Sumo2Unreal. All of the incoming roads have one turn lane and one of the incoming roads is four laned (while the remaining default to two each).	33
3.5	A five way intersection generated using IntGen and rendered on Unreal Engine using Sumo2Unreal.	33
3.7	Decals can be added to the roads as per the connected lanes at the intersection. In this figure, the nearest decals can be interchanged with other relevant decals as shown in Figure 3.8 and in Figure 3.9.	35

3.8	Decals can be added to the roads as per the connected lanes at the intersection. In this figure, the nearest decals can be interchanged with other relevant decals as shown in Figure 3.7 and in Figure 3.9.	35
3.9	Decals can be added to the roads as per the connected lanes at the intersection. In this figure, the nearest decals can be interchanged with other relevant decals as shown in Figure 3.7 and in Figure 3.8.	36
3.10	In Cruzway, roadsplines are generated along with road meshes to communicate road connectivity and other attributes to the vehicles navigating above. Here the road splines have also been annotated with the suggested direction based on lane connectivity sequence.	36
3.11	Road network generated by Cruzway based on a sparse road network generated from TownSim [127].	37
3.12	Procedurally generated navigation bounds volume (green, normal cost) and navigation modifiers (red, high cost) on the four-way intersection. Pedestrians tend to navigate across green areas. . . .	37
4.1	Example QA data	43
4.2	ScenarioQA Overview	46
4.3	An example ontology from the ASAM OpenXOntology standard .	47
4.4	QA Generation	89
4.5	Multi-LLM Evaluation	91
5.1	A circular relationship where effective scenario generation is needed for improving scenario reasoning and effective scenario reasoning is needed for improving scenario generation	124
5.2	Model-based Task Planning Prompting of LLMs. The boxes in blue involve prompting an LLM and obtaining its output.	126

5.3	An example of clustering of scenario contexts for scenarios index range 200 to 400. Each point in the scatterplot represents a scenario context. The color of the point represents the cluster it belongs to with each cluster having a centroid represented by a blue cross mark. We can select a representative set of scenarios based on such a figure and assess diverse scenario generation capability of this system.	130
-----	---	-----

List of Tables

2.1	Summary of scenario abstraction levels	14
2.2	Five Layer Model for AV Test Scenarios. Note: The data formats and tools are not comprehensive and only contain exemplars. . . .	15
2.3	Examples of tools used in scenario-based testing of AVs.	22
2.4	A set of examples regarding the intra-component and inter-component impacts of perspectives within the Small World meta-model pre- sented in this chapter	23
4.1	Evaluation of LLMs for the sets of 10 questions of generated QA data along the pre-defined breadth and depth categories as stated in Section 4.4.	116

Abstract

Controllable and Interpretable Generation of Realistic Autonomous Vehicle Test Scenarios.

by

Ishaan Paranjape

Autonomous vehicles (AVs) need to be rigorously tested for safety before deployment. Scenario based testing has the potential to address this need. The required scale and diversity of these scenarios is large and generators are needed. To determine a sensibility of the range of scenarios needed, in this dissertation, I present a meta-model for organizing important aspects based on perspectives regarding autonomy and scenarios in the literature. Importantly, this aspect organization precedes formalization of scenarios and the corresponding generators. The rules for generating scenarios can either be encoded by a designer or learned from data. Broadly, test scenarios consist of elements with spatial and temporal aspects out of which the most essential elements are roads and vehicles. In this dissertation, I present a designer-encoded high definition road generator, Cruzway RoadGen which is capable of generating roads at the scale of single road intersections or city-sized road networks. While designer-encoded scenario generators have high controllability and interpretability, data-driven generators have the potential of generating realistic and diverse scenarios while reducing the tedium of encoding complex rules. Large Language Models (LLMs), due to their ability to accept a wide range of inputs while being highly generalized data-driven models have the potential for combining advantages of both of these approaches. In this dissertation, I take steps towards realizing this potential. For LLMs to be effective at scenario generation, the challenge of reasoning (and consequently, hallucinations) must be addressed. To address this, I first define a domain-specific concept,

scenario reasoning. I determine a circular relationship where scenario representations consist of System 2 reasoning traces while scenario reasoning problems themselves contain these representations. With CruzwayQA, we design and develop a concepts-first, modular, synthetic QA generation approach to evaluate scenario reasoning capabilities in LLMs. With Cruzway Scenario Generator, we take steps to enhance both scenario generation and scenario reasoning in LLMs, specifically by using prompting and symbolic approaches. With these works, we converge to an effective design a controllable and interpretable LLM-based generator of diverse and realistic scenarios.

Chapter 1

Introduction

The need for rigorous and effective testing of Autonomous Vehicles (AVs), and more generally, robots is an urgent and mostly unsolved challenge despite increasing public deployment over time [57] [58]. The technology that makes vehicles autonomous has increased in complexity over the years as seen in the Baidu Apollo tech stack [14]. The testing of these systems needs to scale in similar proportions. These needs are addressed by *scenario generation methods* within *simulation-based testing*. Here, a scenario is defined as “*the temporal development between several scenes in a sequence of scenes.*” A *scene* is defined as “a snapshot of the environment.” [139]. In the research literature there are five challenges in the field of AV test scenario generation: (1) Fidelity. Fidelity is the *realism* of a scenario since the final objective of testing is a successful deployment of an AV on public roads. (2) Efficiency. This term has to do with selecting and prioritizing scenarios for testing. Since the area of selecting the *safety critical scenarios* scenarios has been covered significantly in the literature [33] [115], in this dissertation, we present approaches that cover the remaining challenges. (3) Diversity. AV test scenarios need to cover *edge cases*. Therefore a generation approach should cover a wide range. (4) Transferability. Several data formats and tools are available

for processing each aspect of a test scenario such as roads and vehicle behaviors. Standardized formats and tool integration approaches are needed for better *interoperability*. (5) Controllability [33]. For the diverse scenario generation capabilities to be realised, a scenario generator’s outputs need to be aligned with user intent. A discussion about *designing* scenarios precedes a discussion about scenario generation. Algorithmic scenario generation assumes that the concept of a scenario is *formalized*, i.e., key ill-defined aspects are well-defined and can be processed. However, in this dissertation, we begin with ill-defined scenarios and detail approaches to shape our perspectives before designing a generator.

Two broad, important factors influencing the generation of scenarios involve: (1) The autonomy feature being evaluated, and, (2) The test scenarios needed for evaluating this autonomy feature. Each factor creates expectations within and with the other factor. For example, end-to-end neural network based AVs create the expectation that the simulation is photorealistic such as in the case of the CARLA simulator [34]. However, if the implementation is a *complete autonomy tech stack* [14], the scenarios need not be photorealistic and can be customized based on the component being evaluated as seen in *TrafficSim* and the *Waymax* simulator respectively [133] [48]. Chapter 2 of this dissertation describes a theoretical framework for determining affordances and constraints due to the consideration of several perspectives (along similar lines as the autonomy perspective stated above) regarding these two factors. Based on these affordances and constraints, key aspects of a scenario can be determined, formalized and optimized. Coverage of these perspectives can result in scenario generators that can generate a *diverse* range of *realistic* scenarios. To have this capability, a scenario generator will need *worldly knowledge* [153]. In this dissertation, I make the claim that Large Language Models (LLMs) have this capability.

LLM-based scenario generation can help synthesize a combination of designer-encoded and learned scenario generation approaches due to their *task generalization* and *worldly knowledge* [153] [113]. This is explored in Chapters 3, 4, and 5 of this dissertation. By definition, designer-encoded scenario generation offers ***control*** while the learned scenario generation offers ***realism or fidelity***. The promise of LLM-based approaches to combine these important advantages into one is presented in Figure 1.1. LLMs are a promising resource due to the following reasons: (1) LLM inputs are open, implying that the *vocabulary* of the input need not be limited to a set of words or data formats. (2) They have demonstrated emergent capabilities for *in-context learning* [145] which allows a user to have the LLM *reason* and thereby control the generation process more effectively, though the generation is still not deterministic. (3) LLMs can interact well with real-world datasets such as AV driver disengagement datasets, traffic accident datasets and traffic rule description datasets since they are all available in natural language. Approaches that analyse and process the same using LLMs are presented by Wang et. al and Guo et. al in the *TARGET* and *SOVAR* projects respectively [143], [32] [49]. (4) LLMs can generate and interpret *functional scenarios* which are defined in natural language thereby affording the generation of *diverse* scenarios. In *LeG-end*, an LLM-based approach to transform functional scenarios to more specific logical and concrete scenarios has been presented [135]. Here, the authors emphasize the need to consider functional scenarios for generation due to the ***diversity*** afforded by them in the corresponding the lower abstractions mentioned above. However, LLMs come with their own limitations. A significant drawback is in its reasoning capabilities [61] [16]. To fully realize effective LLM-based scenario generators, we will first need to measure and improve ***scenario reasoning***. The concept of scenario reasoning is defined in Chapter 2 of this dissertation. Ap-

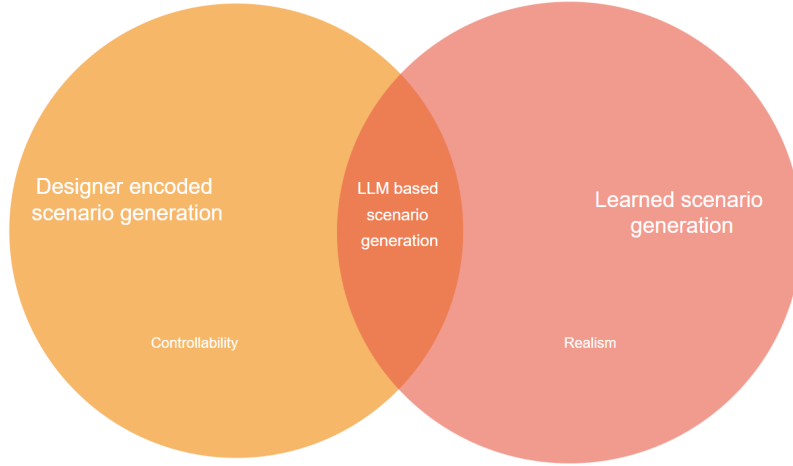


Figure 1.1: The promise of LLM-based test scenario generation

proaches for measuring and improving scenario reasoning in LLMs are presented in Chapters 4 and 5 respectively. Specific research questions and contributions in this dissertation are stated below.

1.1 Research Questions

The following research questions, tagged with the field that the respective answers can contribute towards, are addressed in this dissertation:

1. (Scenario-based testing of AVs, AI Safety) **What are the important perspectives that would precede formalization in the design of an algorithmic scenario generator?**
2. (AI Interpretability, AI Safety) **How do we improve AV test scenario reasoning in LLMs?**
3. (Scenario-based testing of AVs, AI Safety, AI Interpretability) **How do we generate diverse, realistic and interpretable AV test scenarios using LLMs in a highly controllable manner?**

To address these research questions, I make the following contributions:

- A theoretical framework for consideration of several perspectives regarding the nature of autonomy and scenarios such that key aspects of a scenario (to formalize) are determined. This is presented in Chapter 2. With the help of this framework, we develop a list of affordances and constraints on the autonomy and scenarios due to specific perspectives.
- ***Cruzway RoadGen***, a designer-encoded road generator with *Cruzway*. By definition, the HD roads generated are realistic and interpretable. The road generator is highly controllable. An output from this system is presented in Figure 3.5. With Cruzway Roadgen, we generate high definition road intersections as well as town-sized road networks in Unreal Engine 4, a popular choice for AV simulators e.g. CARLA simulator [34]. This is presented in Chapter 3.
- ***CruzwayQA***, a *concepts-first*, modular, synthetic QA generation approach to evaluate reasoning capabilities of LLMs. This is presented in Chapter 4. We generate QA datasets that evaluate scenario reasoning capabilities in LLMs. The questions involve ontology and scenario generation in accordance with the five scenario generation challenges listed in this chapter.
- ***Cruzway Scenario Generator***, a system to generate realistic, diverse and interpretable scenarios using LLMs in a highly controllable manner, specifically by using an *LLM-Modulo* approach. This is presented in Chapter 5. Here, we generate dynamic scenarios in the PDDL format. We address the five scenario generation challenges listed in this chapter.

The rest of this dissertation is divided into chapters as follows; in Chapter 2, we present a framework to determine focal aspects of AV test scenarios considering



Figure 1.2: A five way intersection generated using Cruzway RoadGen.

the nature of autonomy being tested and AV evaluation goals. In Chapter 3, we present Cruzway RoadGen. In Chapter 4, we present CruzwayQA. In Chapter 5, we present Cruzway Scenario Generator. In Chapter 6, we cover a conclusion and future works discussion. Throughout the dissertation, the focus is to help the reader develop an understanding to design scenario generators that accomplish the five challenges listed in this chapter.

Chapter 2

A meta-model for Scenario-based Testing of Autonomous Vehicles

Autonomous Vehicles (AVs) are increasingly gaining adoption across several cities in the world [53]. However, similar to other forms of AI, they need to be comprehensively evaluated for safety. With scenarios, we can test the entire AV (possibly in simulation) for a situation similar to what it might experience in the real world. A significant challenge is that of effective coverage of test scenarios over the real world situations. Automated scenario generation approaches with wide search spaces are proposed to address this [122] [115]. However, this too is insufficient since a test scenario assumes a formal representation and there can be an infinite number of formal representations for a given scenario. However, formal representations are designed by human designers from the first person perspective of an individual evaluating for specific testing objectives. These perspectives *precede* the formalization of a AV test scenario and its corresponding generation problem. Therefore, it is important to have discussion about these perspectives, how they determine the focal aspects of test scenarios. These focal aspects can then be formalized based on which generation algorithms can be designed.

In this chapter, I make the contribution of the **Perspective-based Meta-Model for Scenario Design** as shown in Figure 2.1. Here, the scenario is defined broadly and abstractly as a that which *grounds* the AV and grants it affordances for navigating autonomously in this world. A human designer is seen as external to this world, designing both the environment and the autonomy. An overview of this meta-model is presented in Figure 2.1. The perspective of the external designer is crucial since this will determine the *focal aspects* of the two key components of the meta-model, the AV and the scenario. An example of this impact is presented in Figure 2.3. This determination happens based on affordances and constraints on itself (*intra-component impacts*) and on the other component (*inter-component impact*). For example, the named autonomy behavior of *Cruise Control* is for maintaining the state of the vehicle on a highway such that the driver doesn't need to interact with the foot pedals constantly. This perspective on AV test scenario design creates the intra-component (autonomy) constraint of requiring a vehicle that can drive on one lane and maintain safe distance from other vehicles and the inter-component (scenario) constraint of a highway with high-speed vehicles. Other autonomy perspectives include technology stack affordances such as a camera-only sensor suite and scenario perspectives include pedestrian non-verbal communication or the moral dilemma scenario perspective.

The rest of this chapter is divided into two sections for each of the components: (1) Autonomy. Here I present a variety of perspectives on the notion of autonomy from conceptual to technical perspectives. (2) Test scenarios. Here, I present the representations of a test scenario considered across a representative set of works from the literature of scenario-based testing of AVs. In each of the sections, I present a list of perspectives of the same and how these can influence the focal aspects of both components of this meta-model. A significant takeaway is that

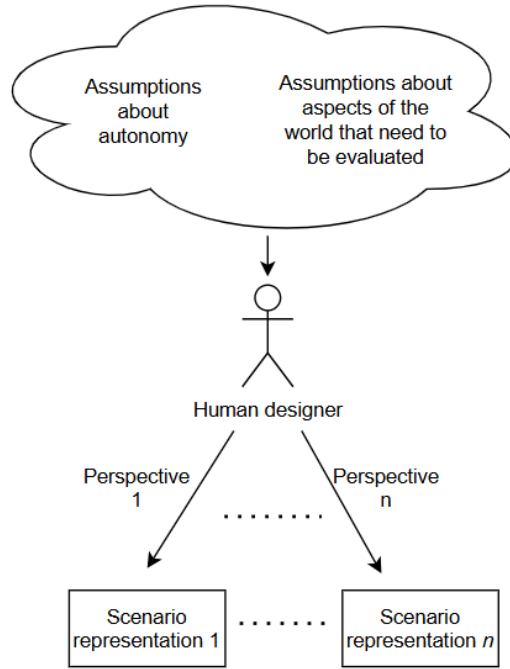


Figure 2.1: A high level overview of the meta-model.

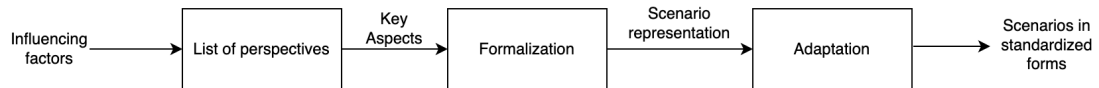


Figure 2.2: Clarifying perspectives helps us identify aspects to formalize for scenario generation using standardized forms.

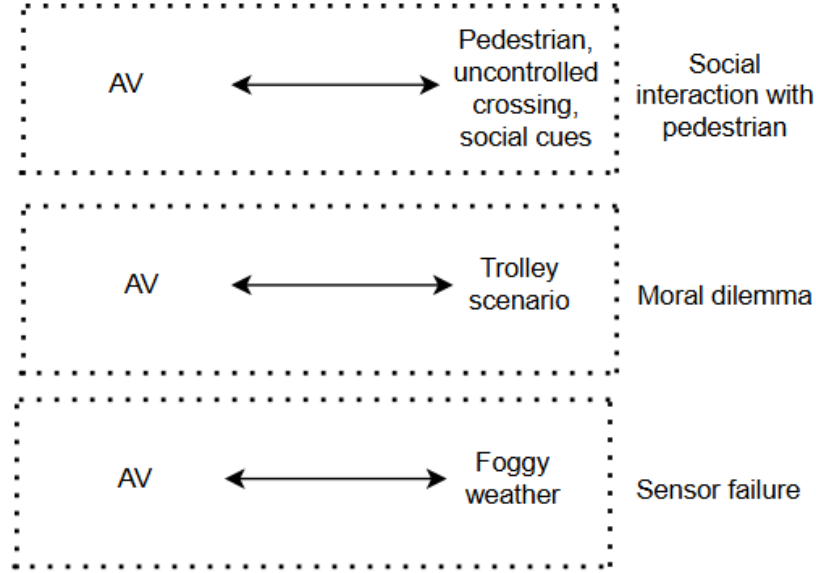


Figure 2.3: Differing perspectives can affect the focal aspects of the two components of the meta-model.

focal aspects are often *relational* and must account for affordances it grants and is granted by other components in this model.

2.1 Perspectives on Autonomy

Generally, an AV is an entity that can carry passengers and autonomously navigate to a destination. The constitution of an AV can differ significantly depending on the human designers involved such as ones from academia or the industry. Based on the works in the research literature, I define the two broad categories of autonomy as behavioral autonomy and technical autonomy. These are covered in the subsection below

2.1.1 Behavioral Autonomy

A typical notion of autonomy is defined in the *5 levels of autonomy* as defined by the Society of Automotive Engineers in the J3016 standard as explained in the corresponding user guide by Koopman [72]. Here, the definition of a behavior can be adopted from planning, where a behavior is *a sequence of conditional actions to go from an initial state to a goal state*. Another perspective on autonomy is from descriptions of named autonomy features in vehicles such as *Cruise Control*, *Automated Emergency Braking* and *Anti-Lock Braking System* [118] [151] [18]. Each conception of autonomy brings with it behavior expectations. A list of behavior expectations are presented in [138].

2.1.2 Technical Autonomy

Beyond the *concept* of behavioral autonomy, I also define the technical perspective of autonomy. At a high level, the implementation can be *modular* or a *monolith*. A good example of the modular stack is the Baidu Apollo AV stack [14] and that of a monolith is the *end-to-end neural network* based implementations. Chen et. al present a survey of research regarding these neural network based monolith implementations [25]. The former is more interpretable than the latter but the latter has the benefit of *joint optimization* of multiple modules to better map the inputs (images of an environment) to the outputs (low level controls for the vehicle such as steer, brake and acceleration). In addition to these two types of implementations, there can also be implementations that are *partially modular* where only specific components are implemented and assumptions are made about the other. Suo et al. in *TrafficSim* present an assumption for their autonomous system that perception-related tasks have been carried out successfully resulting in bounding boxes around entities in the simulation and only controls related tasks

such as behavior prediction and path planning are needed. Each decision above impacts behavioral autonomy and AV test scenario design. These are presented in the subsections below.

2.1.3 Intra-component Impact: Relations between different autonomy perspectives

The impact of the perspectives presented above have a bidirectional relation to other perspectives within the same component. Conceptual autonomy related objectives set *expectations* from the technical implementation and *affordances* are determined in the opposite direction. Behavioral autonomy perspectives create expectations in terms of *object and event detection response (OEDR) capabilities* from the technical stack. This defines *task specific* functional expectations of the technical features such as from an object detector within the technical stack. For example, in the framework presented by Thorne et. al, the L3 automated traffic jam autonomy feature created the expectation that the system will be able to detect interactions such as lead vehicle decelerating, a pedestrian being inside or outside the crosswalk or a cyclist riding in a dedicated lane [138].

2.1.4 Inter-component Impact: Expectations from AV Test Scenarios

The same expectations above also require test scenario conditions the ground the OEDR capabilities. This can be in the form of bike lanes, crosswalks and the corresponding types of dynamic scenario participants being present and demonstrating typical behavior. Further, the scenarios need to be tuned to the type of autonomy being tested as described above i.e. whether the simulation needs to

be photorealistic or having another representation (bounding-boxes) or whether the simulation needs to be physics-based.

2.2 Perspectives on AV Test Scenarios

Several perspectives on AV test scenarios are available in the literature. These include conceptual representations (including standardized data representations), scenarios in simulation tools, scenarios in datasets, and scenarios in generation algorithms. In the subsections ahead, we go over these in detail.

2.2.1 Conceptual Perspectives

Scenarios are the second component of the model presented in this chapter. In accordance with the six layer model [121], broadly, scenarios can have *static* (spatial) and *dynamic* (temporal) aspects and their organization. In the CARLA AV simulator’s Scenario Runner tool [23], scenarios include 3D assets in the game engine along with their organization in the form of behavior trees. The designer’s perspective can influence the AV test scenario as well. This perspective along with the nature of autonomy defined in the previous section influence the focal aspects of the scenario which in turn determines their formalization. For example, we might want to evaluate the response of the Cruise Control autonomy feature in a *moral dilemma* type of scenario. Several scenario representations are proposed in the scenario-based testing literature from varying perspectives [115] [33].

It would be valuable to map the six layer model to the standardized representations of test scenarios. Regarding standardized formats, layers 1-3 are covered by formats such as OpenDRIVE and layers 4-5 are covered by scenario formats such as OpenSCENARIO [93]. Our focus in this chapter is on static scenarios or

Abstraction Level	Representation Type	Tools
Functional	Natural language keyword based	ScenicNL [37]
Logical	Parameter ranges	Scenic [42]
Concrete	Specific parameter values	Cruzway RoadGen [109] [108]

Table 2.1: Summary of scenario abstraction levels

the first three layers. The OpenDRIVE format is a high resolution, XML-based representation of a road network consisting of definitions of roads, lanes (including the sidewalk), lane markings and road objects such as traffic signs and medians. Since traffic lights change their state with time, their definition is partially covered by the OpenDRIVE and the OpenSCENARIO formats respectively. Several alternatives are available to the OpenDRIVE format such as the CommonRoad and SUMO’s road network format depending upon the road design or generation tool being used [5] [84]. Another road network format is the OpenStreetMap (OSM) format [102]. This is a widely used, open, XML-based format for defining roads on the OSM platform. This platform consists of map data from the entire world. The OSM format however, is not as high resolution as the OpenDRIVE format. In this chapter, I separate these types of road networks into two categories: *regular road networks* and *HD road networks*. Since simulation testing of AVs requires diverse, photorealistic scenarios, our focus in this chapter will be on HD roads. Several tools are available for drawing, editing, generating and converting the data formats mentioned above. To prevent reengineering, we can make use of these tools within our automated road generation system. These tools are summarized in the forthcoming section.

The second approach regarding conceptual representations is the *functional-logical-concrete* scenario model where scenarios are represented by their abstraction level. At the highest abstraction level, I define *functional scenarios* in natural

Layer	Definition	Data Format	Tool
1	Roads (static)	OpenDRIVE	Cruzway RoadGen
2	Road Objects (static)	OpenDRIVE	Cruzway RoadGen
3	Road Objects (dynamic)	-	-
4	Dynamic actors	OpenSCENARIO	Scenic / CARLA Scenario Runner
5	Weather	OpenSCENARIO	CARLA Scenario Runner

Table 2.2: Five Layer Model for AV Test Scenarios. Note: The data formats and tools are not comprehensive and only contain exemplars.

language (with a fixed vocabulary or open vocabulary). The next level are *logical scenarios*, defined in terms of parameter ranges. Lastly, I define *concrete scenarios* in terms of specific parameter values. Based on granularity of input descriptions, scenarios are categorized by Menzel et al. [93] into three groups:

1. Functional, which is a verbal description of the scenario. An example of this would be “car A follows car B.” These scenarios are described in natural language.
2. Logical, which is a scenario description in terms of parameter distributions. An example of this would be the distribution of a specific parameter such as velocity of a vehicle following a normal distribution. Scenarios defined in the *Scenic* language are an example of logical scenarios [42].
3. Concrete scenarios, where specific parameter values are provided for a scenario. An example of this would be a road intersection with 3 incoming roads, where each road has two lanes and each road contains one vehicle headed towards the intersection at a velocity of 20 miles/hour. In Cruzway

RoadGen described in the next chapter, I specify scenarios using the concrete granularity level 3.

Tools which can work with scenarios in different forms of description can accept a much wider range of scenario data. For example, a functional scenario generator can be designed to read and generate scenarios from social media datasets or police reports [44]. Tools that convert scenario descriptions from one of these aforementioned forms to another can help bridge scenario generators [93]. I discuss tools for scenario generation in the respective subsection 2.2.5 below.

2.2.2 Scenario Dataset Perspectives

The perspective of test scenarios considered within datasets is that of *vector representations* for the training or testing of data-driven models. Datasets are a common occurrence in the scenario based testing literature due to the presence of black box, data-driven models which significant evaluation test data for verification. Data driven models occur at each module of the technology stack. Broadly, the modules of concern are perception, localization and planning. For perception, broadly, a scenario may involve an image or a sequence of real or simulated images or point cloud representing the output of the most important sensors; cameras and lidars. Regarding localization, the test scenario perspective might involve an image (first person view or birds eye view) which must be transformed into a spatial model. Regarding planning, the assumption is that perception and localization has been carried out successfully and the representation of scenarios for these modules are bounding boxes in 3D space.

A variety of test scenario perspectives are considered directly in the scenario generation algorithms as well. These may or may not match the representations in datasets or the conceptual models discussed above. These perspectives are

presented in subsection 2.2.3 below.

2.2.3 Scenario Generation Perspectives

Several perspectives for test scenarios are considered within scenario generation algorithms as well. I use the categories mentioned in [139] to classify these methods. These include ontology based scenario generation, data driven approaches and testing based approaches. In ontology based approaches, scenes are interpreted according to a framework after acquisition of scene data in a different form. In data driven approaches, important scenarios are extracted from existing datasets using approaches such as clustering or classification. In testing based approaches, scenarios are evaluated based on metrics, the most common of which is criticality. This may or may not include performance of the model AV.

In [13], an ontology based method for scene creation is proposed. In this method, scene related knowledge from different sources such as regulations and guidelines is acquired and assimilated into a layered model as defined in the previous subsection. Then, traffic scenes are generated in an automated manner from this model.

Data-driven approaches include scenario extraction from a database or generation of a database of scenarios. In [71], critical scenarios are generated by first extracting road intersections from an Openstreetmap database. Traffic data from the SUMO traffic simulator is generated on top of this road data. Then the criticality of the scenarios in this simulation is increased to yield desirable results.

Taxonomies are significantly influenced by the AV components under test and definitions considered. Scenario taxonomies are created on the basis of input or content. The kind of content specified is quite similar to content in games. Therefore procedural content generation (PCG) methods for games would be rel-

evant here. PCG would reduce the authorial burden to generate a large number of scenarios and there are subfields in PCG which would require evaluation and selection of content. Evaluation and selection of content is done in the scenario based testing literature as well. Therefore, a literature review of PCG methods relevant to scenario based testing is done in the next section. The goal is to gain insights to move in the direction of designing a procedural scenario generator for testing AVs.

Search based procedural content generation (SB-PCG) for AV test scenarios brings together the fields that have been reviewed above. Here, evolutionary search is used for optimizing the generation of content before it's algorithmically generated. It's been mostly used by games researchers and has also been used by researchers for generating test cases/scenarios for AVs. More research in this topic is required. Our proposed dissertation research addresses this need. This method falls within the category of EAs where a fitness metric is used for searching for a subset of scenarios (usually critical scenarios, ones which cause accidents). Projects using evolutionary algorithms inherently qualify for a project that would generate an emergent result since the rules of a genetic algorithm are quite simple and generate outputs with a complex causality. Search based PCG certainly has immense potential in generating scenarios that belong to the long tail. Below I review some works in generating AV test scenarios using evolutionary search-based methods. Before that, I briefly detour to cover the long tail.

The long tail of AV test scenarios is addressed less compared to ideas related to critical scenarios. Improving predictions for the long tail is addressed in [89] where implicitly the long tail scenarios are classified as rare and challenging. Long tail scenarios are separated from easy scenarios in feature embedding space by using contrastive learning. In the proposed project a much simpler heuristic is used;

clustering to identify the rare scenarios.

In [?], a multi-objective search is used to generate AV test scenarios. The objectives are number of collision and potential collision scenarios, scenario diversity, scenario demand and attribute prioritization. This scenario representation and fitness function are integrated with several multi-objective search algorithms and compared based on a quality indicator from [?]. The drawback in this work is the simple genome provided, where each scenario is a list of parameters. More details about the proposed project are provided in the forthcoming chapters.

In [77], a genetic algorithm for AV testing, SceGene is presented. A hierarchical genetic encoding of a scenario containing roads and traffic participants, specifically their motion trajectory sequences (as a sequence of coordinates) followed by events inferred from these trajectories is shown here. Four macroscopic properties are provided here for controlling the genotype to phenotype mapping. Recombinations and mutations address a layer in this hierarchy such as switching trajectory sequences and changing destinations of individual trajectories respectively.

2.2.4 Scenario Evaluation Metrics

Since the goal of generating scenarios is testing AVs, each scenario once defined and generated needs to be evaluated for its ability to test for the safety of an AV. In the literature scenarios are prioritized based on *criticality*. In [147], criticality is defined as “the risk of the nearby vehicles (to the ego vehicle) if the current traffic situation continues.”

In testing based approaches, criticality metrics are used to identify important scenarios. We can loosely define these metrics as the ones which determine whether vehicles will collide with each other or with pedestrians. In [139], we see

Key Performance Indicators (KPI) to evaluate AV performance. Here, Time-to-Collision (TTC) based metrics are the most commonly used. TTC is defined as the time period it would take for two vehicles to collide provided they continue on the same path at the same speed. Other metrics can also be used such that optimizing them increases scenario criticality. One such metric is the drivable area, which is the space of possible trajectories that an AV can take considering the road geometry, other vehicles and obstacles [128]. Reducing the drivable area increases scenario criticality.

The notion of critical scenarios and using collisions as a measure of criticality percolates into works in this subfield as well. In [?], fitness function templates are provided for researchers who would use search while generating scenarios for testing AVs. Safety distance is a metric for evaluating ego vehicle safety. Interestingly, this paper suggests *form templates* which are characteristics of non ego vehicles to make an accident more likely; a vehicle in front of or very close to the ego vehicle while it's driving forward at an intersection would make the scenario relatively more dangerous than a vehicle driving behind the ego. We make use of these templates in our fitness function proposal. In [?] and [?] a search is carried out for avoidable collisions, ones which could be avoided if an alternative configuration of an AV was used. The drawback of this approach is that it tests only a path planner.

2.2.5 Simulation Tool Perspectives

Several tasks can be carried out to work with HD roads such as drawing them on an Graphical User Interface (GUI), editing them on the same, generating them by means of an algorithm or converting/transforming them to or from another standardized format for *interoperability* among tools. Within the SUMO traffic

simulator’s road network tools, we can draw and edit road networks using *Netedit*, generate roads using *Netgenerate* and convert roads to or from other formats such as OpenDRIVE using *Netconvert* [84]. These tools require us to use SUMO’s own xml-based road network format.

Regarding the OpenDRIVE format, we can make use of the *Blender Driving Scenario Creator* [120] to draw roads in this format. Alternatively, we can make use of *Roadrunner* within MATLAB to draw the same [90]. However, unlike the former, a paid MATLAB license is required for the latter. To generate roads in this format, we can make use of the *scenariogeneration* Python library [7]. Since the Blender Driving Scenario Creator tool is a plugin within Blender which is a 3D modeling tool, we can also create 3D environments in the FBX format. In fact, the CARLA simulator requires that imported maps be in the OpenDRIVE and FBX formats. 3D environments may contain visual information regarding roads (which is missing from HD road formats) such as the texture of the road surface and lane markings. Visual information is necessary to create scenarios specifically to evaluate the perception component of an AV such as puddles of water in the corner of the road after rainfall or significant degradation of the surface and lane markings to name a few. In this project, we integrate several of these tools to generate HD roads that fulfill the requirement of the standardized formats as well as ones which are visually photorealistic such that they can be utilized in a simulation tool for testing an AV. In the next section, we present and discuss a few simulators that are available.

AV simulators can be categorized based on the parts of the AV *stack* that they can test (present diagram) i.e. if the complete AV technology stack is tested, we need photorealistic environments with realistic physics. However, if only the planning and control modules are being tested, we can relax the requirement of

Tool name	Tool type	Inputs	Outputs
scenariogeneration	Python library	Program	OpenDRIVE or OpenSCENARIO file
Netgenerate	CLI	CLI parameters	SUMO .net.xml roads
Blender Driving Scenario Creator	Blender plugin	GUI	OpenDRIVE, OpenSCENARIO and FBX

Table 2.3: Examples of tools used in scenario-based testing of AVs.

photorealistic visuals as presented in TrafficSim [39] or Waymax by Waymo [48]. Here, simulations consist of the *output* of a perception of modules of an AV - annotated bounding boxes moving on a sparsely drawn 2D map (consisting of line annotations denoting a map). Several open sourced, realistic AV simulators are available such as CARLA [35] and LGSVL [116]. The focus of this chapter is to present an approach to generate HD roads for the CARLA simulator or independently in Unreal Engine 4. This simulator makes use of the OpenDRIVE file format for loading data regarding the logical connections of roads and data in the FBX format for the 3D model concerning the roads along with any aspect of the environment. The CARLA simulator is chosen because it is open source and is capable of evaluating a complete AV stack such as Apollo.

2.2.6 Intra-component Impact

Each AV test scenario perspective creates affordances and constraints regarding the use of other perspectives. Multiple perspectives can overlap and co-exist in a system. Conceptual perspectives are highly *interpretable and controllable* and can help designers create customized scenarios. An example of these are Scenic scripts or scenarios defined in Cruzway Roadgen. However, these scenarios are not realistic and lack expressive range e.g. the Scenic language only allows the definition of primitive driving behaviors and not complex behaviors. In addition,

scenario descriptions cannot be scaled. A standardized representation of a single four-way road intersection in the high definition OpenDRIVE format results in large file sizes and cannot scale well to city-sized maps. While the OSM formats do allow scale, they lack the high-resolution definitions needed for AV testing.

Dataset-based scenario perspectives have the opposite issue. While they *realistic*, they cannot be *controlled or interpreted* easily.

2.2.7 Inter-component Impact

Scenario perspectives considered require the autonomy being tested to have that capability. An example of this is the object and event detection capabilities (OEDR) such as a safe response to the *lead vehicle decelerating* vehicle behavior in a scenario expected for the *L3 conditional automated traffic jam feature* as shown by Thorn et. al [138]. Such capabilities can be informed by the designer intent. The type of autonomy being tested needs to have the scenario perspective within its Operational Design Domain (ODD). For example if the designer has an intent to design pedestrian social cues type of scenarios, the autonomous vehicle needs to have a features designed and developed to be capable of recognizing social cues or respond to it or both.

Perspective on Autonomy	Constraints on Scenarios	Perspective on Scenario	Constraints on Autonomy
Cruise Control	Highway	Moral Dilemma	Ability to make moral decisions
Automated Parking	Street Parking or Parking Lot	Social interaction with pedestrian	Perceiving social cues

Table 2.4: A set of examples regarding the intra-component and inter-component impacts of perspectives within the Small World meta-model presented in this chapter

2.3 Conclusion

In this chapter a novel meta-model for scenario perspectives was discussed. Perspectives is an initial designer intent for the scenario based testing of AV tasks e.g. safe AV responses to pedestrian social cues, moral dilemmas and more. Perspectives determine focal aspects of two key components of a scenario-based testing model; autonomy and scenarios. Both components can take various forms and focal aspects of one can influence the other. In addition there can be multiple focal aspects within one component; they can influence each other as well. Determination of these focal aspects precedes formalization of the scenario based testing and/or automated scenario generation problem. In this chapter, we determine a list of designer intents that can be considered and the corresponding focal aspects for each intent. An understanding of these perspectives broadens our understanding of scenarios that are truly relevant to AVs. Current automated scenario generation approaches fall significantly short of this and we need better paradigms to reach this scale of scenario generation. Through my work in future chapters, I claim that LLM-based scenario generation approaches are more promising for scaling up scenario generation to this level. However, they have problems of their own such as hallucinations and reasoning challenges which need to be addressed.

Chapter 3

Cruzway RoadGen: Designer-Encoded Realistic Road Generation

3.1 Introduction

A large number of scenarios are required to comprehensively evaluate an AV for safety. This is usually highlighted in terms of miles driven by the AV in simulation and in real world testing. To scale the test scenarios in simulation, we need scenario generation algorithms. Due to the presence of perception components in the AVs, environments considered for simulation testing often resemble environments set up for photorealistic video games. Therefore game engines are often used for AV simulation testing. Some of the major AV simulators such as CARLA and LGSVL are built using game engines such as Unreal Engine 4 and Unity [34]. A key aspect of these simulation scenarios are road networks. Designing and developing a photorealistic road network generator for simulation testing is especially

challenging due to the following reasons: (1) Road networks are a form of logically connected graph involving road intersections and driving lanes with or without sidewalks. Photorealistic realisations of these road networks typically also contain lane markings i.e. signs that describe these logical connections. (2) Photorealistic 3D road meshes have a variety of visual properties reflecting these connections. This includes the color and texture related attributes e.g. sidewalk lanes appear different than driving lanes or geometry attributes e.g. turn lanes emerge from a driving lane near an intersection if a turn in that direction is allowed for vehicles.

In this project, Cruzway RoadGen, I procedurally generate 3D road meshes in the game engine, Unreal Engine 4, the 3D modeling tool Blender and an AV simulator, CARLA simulator. Multiple tools are build to support and facilitate road generation processes. Our work in this project affords easily increasing the types of roads that can be used for simulation testing of AVs at city resolution and at the resolution of individual intersections.

Cruzway RoadGen is a system that generates visually and logically realistic roads for AV testing. This system consists of several pipelines of tools. It would be valuable to reflect on design decisions taken within the greater research landscape surrounding scenario based and simulation based testing of AVs. We briefly discuss this literature here as needed for clarity. It is important to note that the focus in this chapter is on the generation of static scenarios, specifically roads and adjacent aspects of the 3D testing environment that do not change with time.

For this project, we consider systems that generate scenarios for testing a complete AV stack (akin to testing the Apollo AV in the CARLA or LGSVL simulators). To this end, we automatically generate visually photorealistic roads and logically complete roads in standardized formats. We generate roads in two key formats: SUMO traffic simulator’s *.net.xml* format and the *OpenDRIVE* format.

Making use of the former grants us the ability to use SUMO’s road generation and viewing tools such as *Netgenerate* and *Netedit* respectively. Generating roads in OpenDRIVE grants us the ability to instantiate these environments in the CARLA simulator and carry out AV simulations. In the next section, we describe our system in detail.

This project involves road generation tools which are supported by several external tools. The tools built within this project include the following: (1) IntGen: This tool generates basic data for a single road intersection and is available as a CLI application. (2) Sumo2Unreal: This tool helps transform road network data into photorealistic 3D road meshes in Unreal Engine 4. (3) BlenderOpenDRIVEImporter: Another road data importer, this time transforming OpenDRIVE HD roads into 3D meshes in Blender. The external tools within this project are: (1) The SUMO traffic simulator’s *Netgenerate* tool. (2) Town-sim-py, an evolutionary algorithm based city generator (add citation). (3) IntGen, a tool to generate data for a single road intersection (add citation). All of these tools are organized into pipelines and systems for easily generating photorealistic 3D roads within the Unreal Engine 4 and the CARLA simulator.

The rest of this chapter is divided into sections as follows: In Section 2, we go further into the implementation details of the Cruzway RoadGen tool, specifically each component. In Section 3, we present qualitative results. In Section 4, we conclude and provide ideas for future work based on these projects.

3.2 Cruzway

A hierarchical road network specification file is generated using either *Town-SimPy* or *IntGen* applications for the purpose of generating town scale road networks or individual road intersections respectively. The road network file has a

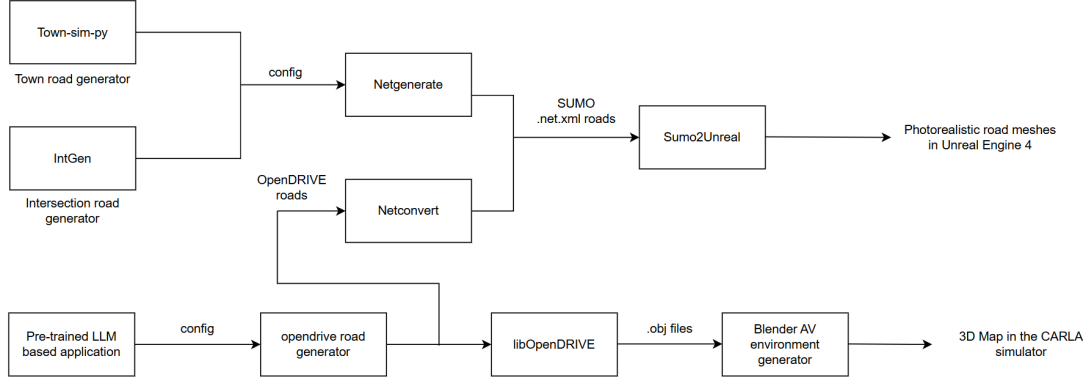


Figure 3.1: The system architecture of CruzWay RoadGen

high-level description of roads, including road length and connectivity information but lacking exact shape descriptions of road and intersection polygons. This specification is then read by a customized *Netgenerate* application (which belongs to the SUMO traffic simulator’s suite of applications) which generates a much more detailed road network file in the standard SUMO road network format. This file describes coordinates of the road, intersection polygons, crosswalks, sidewalks, etc. This file is then imported and converted into a geometry of road meshes into the Unreal Engine using the *Sumo2Unreal* application. A second stage adds procedural navigation meshes (navmeshes). Using this road and navmesh information, other agents (such as NPC cars and pedestrians) are procedurally placed on top of these meshes according to a scenario description. The medium of communication of road data to vehicles and pedestrians in the simulation is through splines and navmeshes respectively. With CruzWay, we can run a wide variety of procedural simulation scenarios to capture many of the situations which an AV would encounter in the real world.

Sparse Road Generation: Townsim and IntGen

TownSim [127] is an agent based city evolution algorithm that generates road networks. It procedurally generates town-sized road networks with different road sizes that vary from grid-like sections to organic, dendritic sections. Figure 3.11 shows an example of a road network generated using TownSim and imported via Sumo2Unreal.

IntGen [105] is a road intersection generator with the capability to create a detailed specification file of a road intersection. The inputs to this application are a number of command-line parameters detailing the intersection information as well as details of individual roads. The output is a road specification file represented in JSON format which can be read by *Netgenerate*. It details the high-level intersection information in the form of the Cartesian coordinates of the intersection, the number of incoming roads for it and the presence of pedestrian crosswalks. This data also specifies details of each incoming road, including the length (in meters), start and end points, the number of lanes (and turn lanes) and the presence of sidewalks. Having these many parameters allows this application to express a large range of road intersection types. Consequently, this would allow an even larger space of scenarios. Examples of four-way and five-way intersections generated by IntGen are shown in Figures 3.4 and 3.5.

3.2.1 High Definition Road Generation: Enhanced Netgenerate

Netgenerate is a road network generation tool within the SUMO traffic simulator suite of applications. Netgenerate typically has the capability to generate synthetic grid networks, spider networks and random networks. For CruzWay, we extended Netgenerate [106] to accept the JSON formatted road specification files

created by *IntGen* or *TownSimPy*. The output of Netgenerate is a road network XML file in the SUMO road network format. This XML file contains much more detail regarding individual roads and sidewalks. By extending Netgenerate, we can generate a wider variation of SUMO road networks than the three mentioned above. We can then use other SUMO applications such as *Netconvert* which can convert SUMO road networks to other standard road network formats such as OpenDrive and MATsim. This pipelining allows us to generate in multiple road network formats thereby allowing us to generate roads for multiple platforms. It also permits running traffic simulations using SUMO. We can then import this road network into the Unreal Engine as procedurally generated road meshes by using *Sumo2Unreal*.

3.2.2 Realistic Road Generation: Sumo2Unreal

Sumo2Unreal [107] is an Unreal Engine plugin for importing a SUMO road network file into the Unreal Engine as procedural road meshes. Unreal Engine is a widely used game engine that supports the creation of real-time software with high-fidelity graphics. Procedural road meshes generated include driving lanes, sidewalks and pedestrian crossings. Using Unreal Engine provides us with many useful capabilities, such as built-in 3D models such as cars, an API for creating BTs, along with photorealistic graphics such as a variety road textures. Sumo2Unreal generates stop signs for road networks describing road intersections only (such as the ones from IntGen). Sumo2Unreal also generates splines from a collection of SUMO road way points. These splines can be used by BT controlled cars to follow logical lanes and make turns correctly. Splines also contain other information, such as other connections with other splines (in the driving direction) and whether there is a stop sign at the end of the spline segment. Together, the

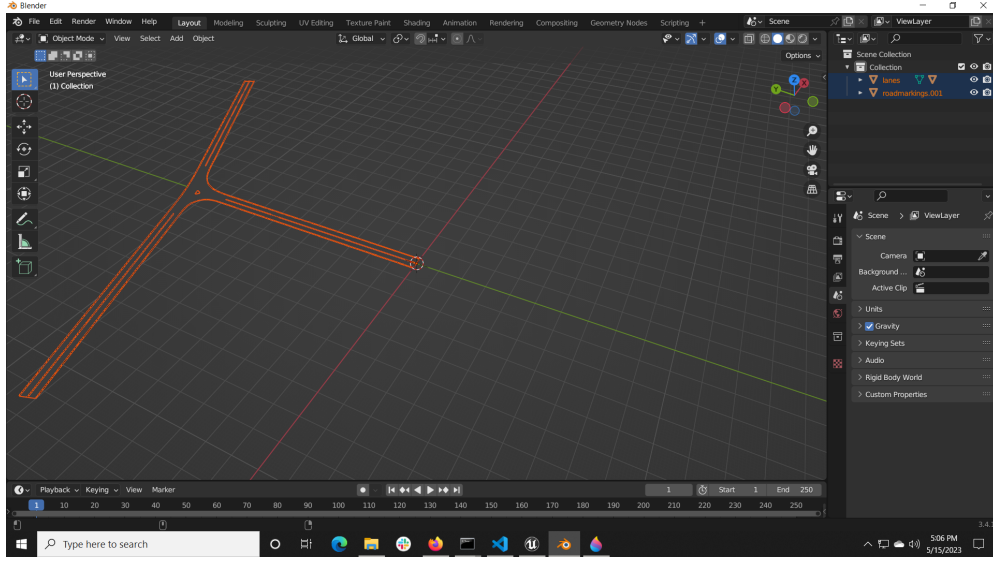


Figure 3.2: A road intersection generated by blenderOpenDRIVEImporter

splines and the navigation meshes are able to communicate the map data to cars and pedestrians in the simulation.

3.2.3 Blender OpenDRIVE Importer

An alternative to the Cruzway approach presented above is using the blenderOpenDRIVEImporter. Here, high definition roads are generated in the *OpenDRIVE* format. These roads are then imported into blender as *sparse 3D models* which are then refined within Blender. Blender provides libraries for adding 3D scene elements that are outside the scope of OpenDRIVE, yet are relevant to autonomous vehicle test scenarios such as trees, buildings, and more. Trees often obstruct important road signs. This affords us the ability to generate roads compatible with the CARLA simulator in terms of accepted formats; FilmBox (FBX) and OpenDRIVE (XODR).

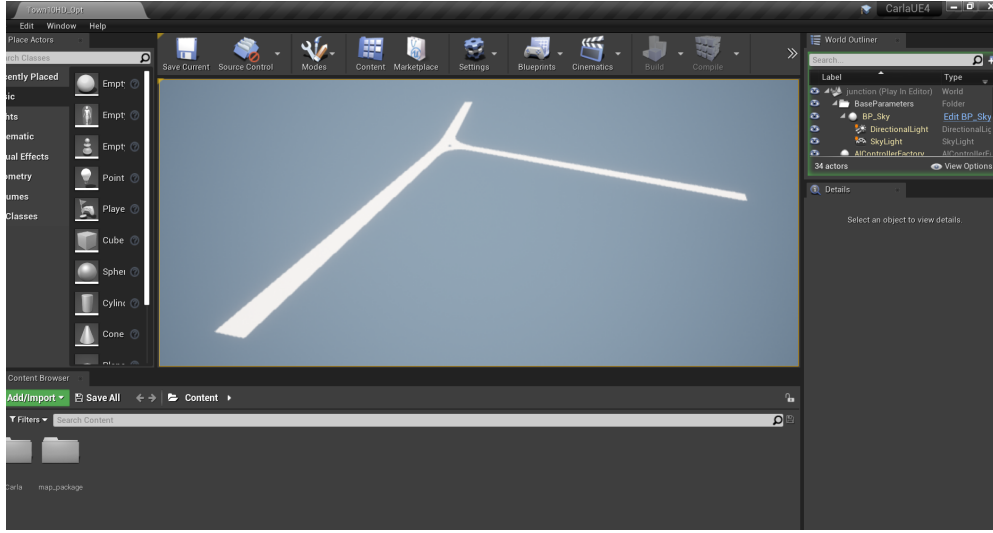


Figure 3.3: A road intersection generated by blenderOpenDRIVEImporter imported into the CARLA simulator

3.3 Evaluation and Results

In this section, I present the road generation capabilities of Cruzway Road-gen. With this system, we can generate individual road intersections as well as town-sized road networks. In the figures 3.4 and 3.5, we can observe high resolution generation of four-way and five-way intersections respectively. Here, the road geometries and textures are realistic. The procedurally generated roads contain turn lanes, pedestrian crosswalks, driving lanes, sidewalks and stop signs. The presence of splines (seen as lines on the roads) serve as 3D objects that can communicate road connectivity data such as *intersection connected lanes* to the vehicle and pedestrians in the simulation. The turn lane signs are chosen based on road connectivity information in these splines. Multiple appropriate turn lane signs can be added for the same location in the intersection as shown in Figure 3.7, Figure 3.8 and Figure 3.9. The roads also ground cost based pedestrian navigation algorithms with the help of *Navmeshes* as shown in Figure 3.12.

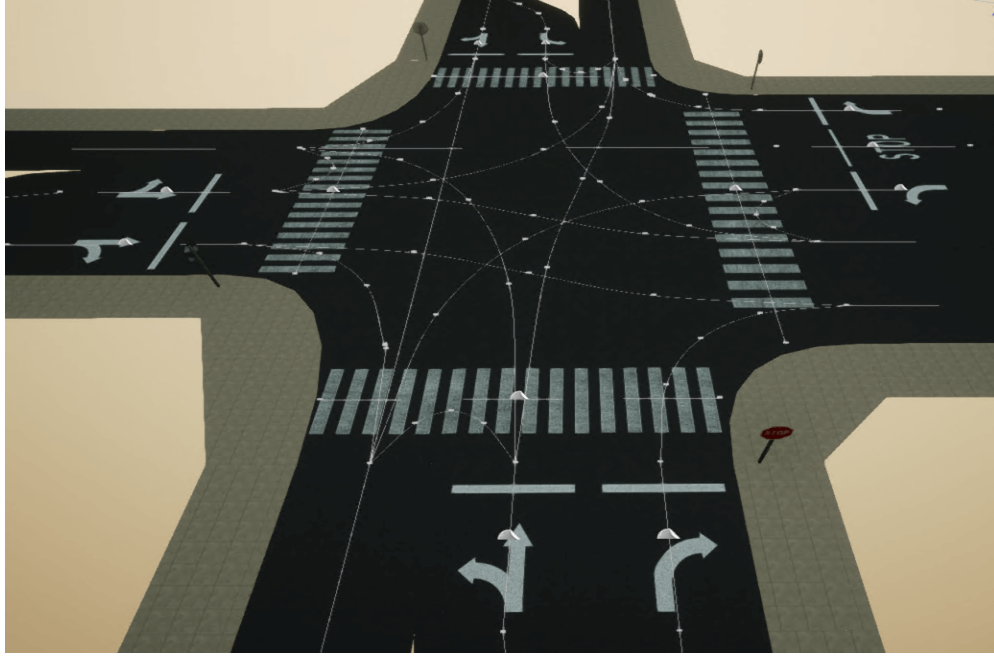
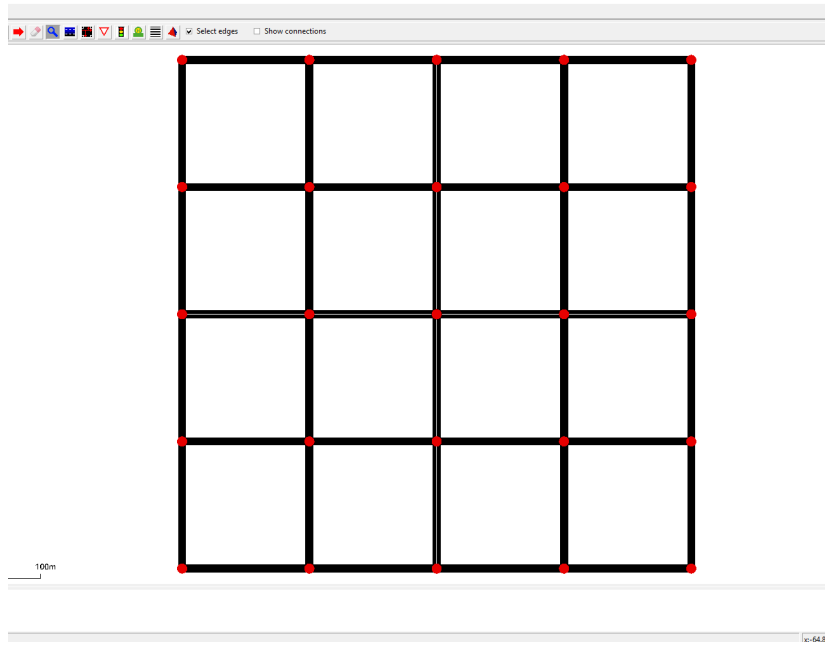


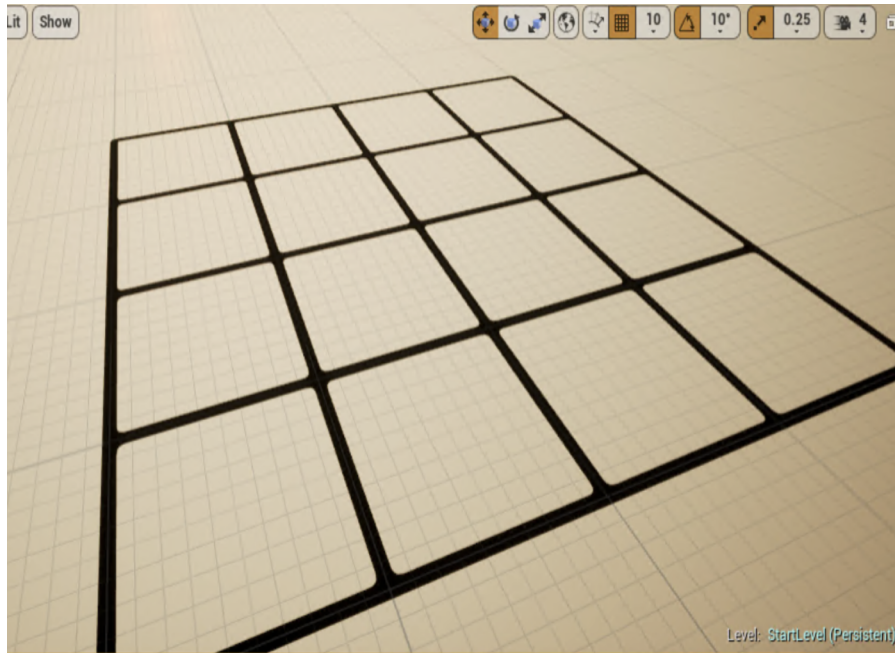
Figure 3.4: A four way intersection generated using IntGen and rendered on Unreal Engine using Sumo2Unreal. All of the incoming roads have one turn lane and one of the incoming roads is four laned (while the remaining default to two each).



Figure 3.5: A five way intersection generated using IntGen and rendered on Unreal Engine using Sumo2Unreal.



(a) A grid network generated by the SUMO Netgenerate tool and visualized in the SUMO Netedit tool.



(b) A grid network generated by Cruzway.

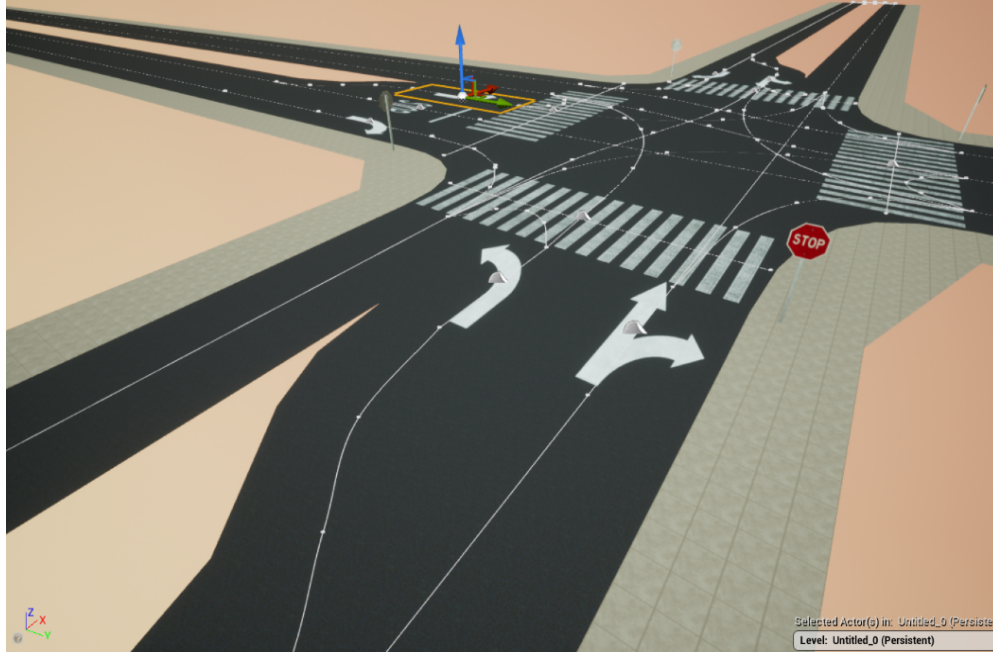


Figure 3.7: Decals can be added to the roads as per the connected lanes at the intersection. In this figure, the nearest decals can be interchanged with other relevant decals as shown in Figure 3.8 and in Figure 3.9.



Figure 3.8: Decals can be added to the roads as per the connected lanes at the intersection. In this figure, the nearest decals can be interchanged with other relevant decals as shown in Figure 3.7 and in Figure 3.9.

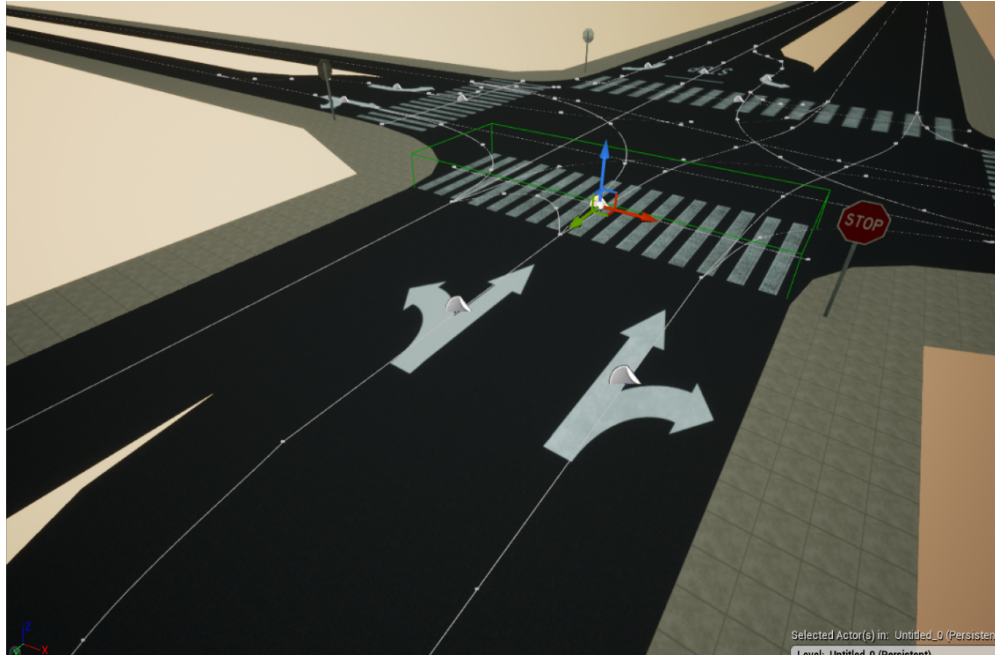


Figure 3.9: Decals can be added to the roads as per the connected lanes at the intersection. In this figure, the nearest decals can be interchanged with other relevant decals as shown in Figure 3.7 and in Figure 3.8.



Figure 3.10: In Cruzway, roadsplines are generated along with road meshes to communicate road connectivity and other attributes to the vehicles navigating above. Here the road splines have also been annotated with the suggested direction based on lane connectivity sequence.

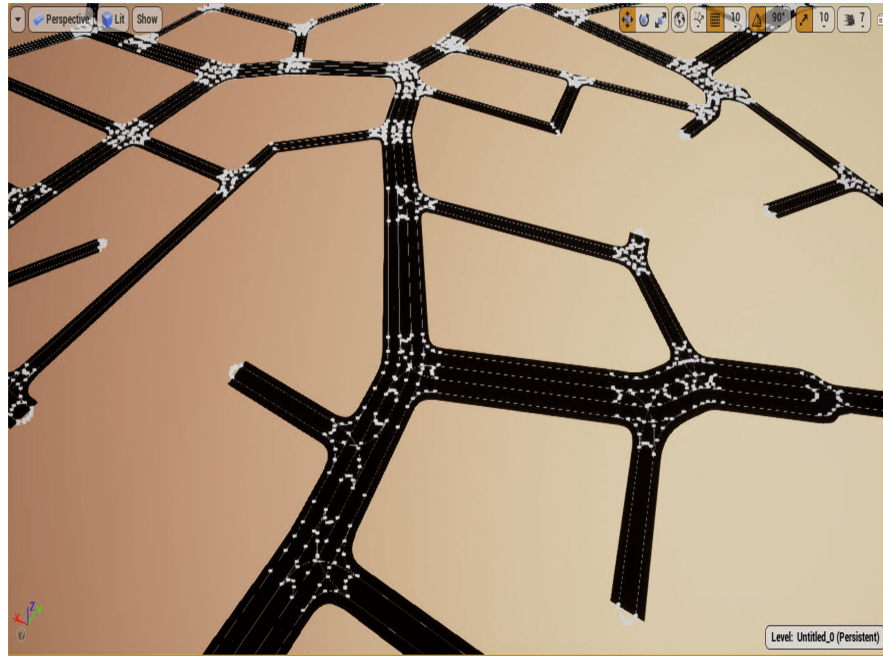


Figure 3.11: Road network generated by Cruzway based on a sparse road network generated from TownSim [127].

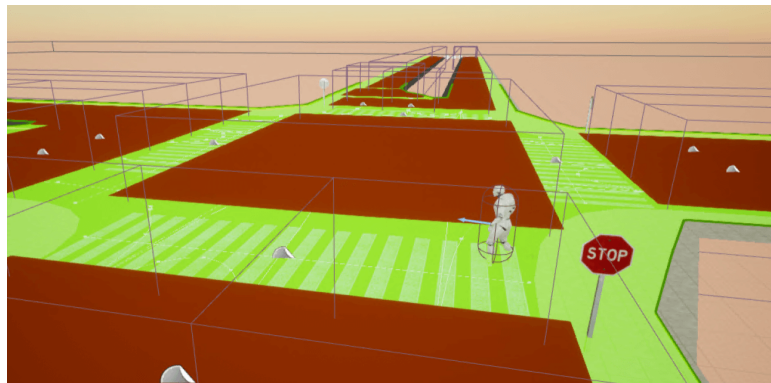


Figure 3.12: Procedurally generated navigation bounds volume (green, normal cost) and navigation modifiers (red, high cost) on the four-way intersection. Pedestrians tend to navigate across green areas.

3.4 Limitations

There are two types of limitations in Cruzway Roadgen: (1) Expressiveness. Roads can be expressed at the resolution of one intersection or a town sized road network. It lacks intermediate levels of resolution for expressiveness. Since the system involves a transformation of Netgenerate road networks to textured meshes in the Unreal Engine, even increasing the parameter values too much can result in overlapping meshes. (2) Tediousness: The effort needed to increasing expressiveness of the generator is quite significant. (3) Realism: The generated roads, while photorealistic and valuable for evaluating perception systems of AVs, are synthetically generated and are disconnected to real-world data.

3.5 Conclusion

In this chapter, I presented Cruzway Roadgen, a designer-encoded static scenario generation approach. Cruzway Roadgen can generate road networks at the resolution of a single intersection or town sized road networks.

Chapter 4

CruzwayQA: Benchmarking Scenario Reasoning Capabilities in LLMs

4.1 Introduction

LLM-based scenario generators can be considered as a *hybrid* form between designer-encoded and data-driven generation approaches in terms of controllability and realism. In the previous chapter, we looked at *Cruzway RoadGen*, where a designer-encoded static scenario generation approach was presented. With *hand-authored* rules, high definition roads were generated in Unreal Engine. Alternatively, rules are automatically modeled from data in data-driven approaches e.g. *ScenarioDreamer* [117]. Designer-encoded approaches offer *significant control* over the generation of content in the form of rules but can be tedious to design and build. Data-driven approaches do not offer this kind of granular control and the only potential approach to modify the generator is to change the training data

itself. Significant research has been done to enhance generation control, such as *language guidance* in image generators [155]. The word *guidance* highlights that this control is not of the same nature as hand-authored rules.

Due to LLMs being uniquely data-driven yet potentially highly controllable, the possibility to develop a controllable LLM based generator that can generate interpretable and realistic scenarios opens up. Large Language Models (LLMs) are *transformer* architecture [142] based natural language generation models. These models are data-driven. Due to their ability to accept a large number of input *tokens* (from a few thousand to around a million), this guidance or prompting in the context of LLMs can be considered to be *dense* [160]. In addition, researchers have been considering LLM *prompting* as a form of programming [40]. LLMs have demonstrated emergent capabilities for *in-context learning* [145] from examples which allows a user to learn based on examples within the context itself similar to training but without any changes to the weights of the neural network. This ensures that LLMs generate outputs which are more aligned with the user intent [103] thereby increasing controllability.

To enhance the potential for realistic scenario generation, we can consider making use of this enhanced controllability and connect the LLM with data and concepts. Several real-world datasets such as AV driver disengagement data, traffic accident data and traffic rule descriptions are available in natural language and can be understood by LLMs as presented by Wang et. al. [143], in *TARGET* [32] and *Sovar* [49] respectively. At a conceptual level, the *functional scenario* abstraction from the scenario-based testing literature is defined in natural language. In *LeGend*, an LLM-based approach is presented to transform functional scenarios to more specific *logical* and *concrete* scenarios [135]. Here, the authors consider functional scenarios due to the *diversity* afforded by them. Since functional scenarios

are considered *informal* and logical and concrete scenarios are considered *formal*, this transformation grants LLMs the affordance to interact with simulation tools such as Scenic [42], a probabilistic language to express *logical* scenarios. LLMs generate Scenic scripts in ScenicNL [37] and ChatScene [156]. A drawback of these approaches is the lack of complex scenarios (specifically driving behaviors) due to the capabilities of Scenic. To support the generation of complex behaviors though, the LLM reasoning capabilities must also match that capability. Current LLM reasoning capabilities are quite low as observed in several QA benchmarks [16]. Further, LLMs have the challenge of hallucinations which further impact reasoning capabilities [61]. These issues must be addressed before the application of LLMs towards realistic scenarios can be considered.

The concepts of hallucinations and reasoning have a significant overlap. Hallucinations in LLMs can be defined as generating non-factual content in the generation process. LLM Reasoning is defined as sequential predictions made by an LLM to draw a conclusion. Reasoning is often connected to the concept of a *world model* which can be defined as a *set of state transitions* [52]. In this paper, we define scenario reasoning as *a synthesis of a non-exhaustive subset of reasoning structures such as deductive, inductive, spatial, temporal as well as System 1 and 2 reasoning with concepts relevant to AV test scenarios such as the ones in the ASAM OpenXOntology*. LLMs are severely lacking in their ability to carry out scenario reasoning []. In [129], an evaluation of multimodal LLMs (ones that can process natural language and image inputs) is carried out as world models for autonomous driving. In this work, while multimodal LMs work well for interpreting individual images, they perform poorly in understanding the dynamics of a scenario across several frames e.g. several LLMs are biased towards identifying motion to be in the forward direction. More effective evaluations are needed to

better understand test scenario reasoning challenges in LLMs. It’s important to note that since we are resolving issues for LLMs to be used as scenario generators, we are considering the reasoning capabilities of LLMs as external observers to an AV scenario occurring on a road. On the other hand, several papers evaluate LLMs’ reasoning from the first-person perspective of the ego vehicle [152]. This requires the LLM to be responsible for perception as well as reasoning about the perceived objects (which are the areas of behavior prediction and planning from the AV literature). In our work, we restrict the reasoning evaluation to be with respect to scenarios described in the narrative form for simplicity. A similar QA generation approach with and for multimodal LLM evaluations is suggested for future work.

Scenario Reasoning

A synthesis of a non-exhaustive subset of reasoning structures such as deductive, inductive, spatial, temporal as well as System 1 and 2 reasoning with concepts relevant to AV test scenarios such as the ones in the ASAM OpenXOntology.

In this chapter, we present *CruzwayQA*, a *concepts-first*, synthetic QA dataset generator that generates multiple-choice scenario reasoning questions from a given scenario. These scenarios are generated from scenario ontologies and an understanding of various reasoning types. To avoid grammatical complexities of string generation approaches, we make use of pre-trained LLMs to generate multiple-choice questions. The structure of questions is inspired from the following works: (1) multiple choice question (MCQ) generation by LLMs by Doughty et al. [36], (2) a description of reasoning types in QA benchmarks such as CommonsenseQA [134] and ScienceQA [85], and (3) OpenXOntology, an ontology for AV test sce-

Scenario:

You are driving an autonomous vehicle on a wet, rainy night. The road is a two-lane highway with a speed limit of 60 mph. Suddenly, a pedestrian walks onto the road from the pedestrian crossing without looking. Your vehicle's safety features kick in and the vehicle automatically applies brakes to avoid a collision.

Questions:

1. (Understand, Causes, 1 Hop)
 What can be the immediate result of a pedestrian suddenly walking onto the road from the pedestrian crossing in front of an autonomous vehicle?
 A. The vehicle will increase its speed.
 B. The vehicle will automatically apply brakes.
 C. The vehicle will change its route.
 D. The vehicle will continue at the same speed.
 Rationale: Understanding the cause and effect relationship and autonomous vehicle's safety features, it's clear that the vehicle will apply brakes to avoid a collision.
 Answer: B

2. (Apply, Causes, 2 Hops)
 If the road surface is slick due to rain, what can be the result of an autonomous vehicle suddenly applying brakes?
 A. The vehicle can skid.
 B. The vehicle will increase its speed.
 C. The vehicle will stop immediately.
 D. The vehicle's lights will turn off.
 Rationale: Applying the principles of physics and understanding that sudden braking on a wet road can lead to skidding, we can infer the most likely outcome.
 Answer: A

Figure 4.1: Example QA data

narios [10]. In the synthetic QA generation process, we ensure that the LLMs are as close as possible to template based generators i.e. they are used for generating coherent questions. We attempt to use as less of the internal knowledge of the LLM as possible so as to avoid a *circular evaluation*. Several variations on the question structure and prompting are carried out to converge onto an effective QA data generation process.

In the development of CruzwayQA, several prompt engineering insights were gained: (1) The quality of the in-context examples significantly determined the questions adhering to specific reasoning categories. (2) The temperature parameter available in LLMs affected the overfitting to the in-context examples. (3) The addition of Bloom’s taxonomy terms combined with the in-context examples increased the *depth* at which the QA samples adhered to a specific reasoning category. (4) Explicitly mentioning the reasoning hops needed in a question did not work and it was more productive to have the reasoning hops be implicitly present in the in-context examples.

The work here provides insights in using LLMs for the creation of synthetic

QA datasets for evaluating reasoning ability of LLMs while avoiding the issue of a circular evaluation by ensuring that the LLM knowledge isn't considered for QA generation. Due to a concepts first, modular prompt being used for QA data generation, we can easily switch modules to alter the domain being considered. Since the dataset is synthetic, significant customizability is possible and significant evaluations over very narrow, niche subdomains is possible.

The rest of this chapter is divided into sections as follows: Since the concept of reasoning is convoluted and unclear, we define the concept of scenario reasoning considered in this chapter in Section 4.2. The method followed for QA generation and the LLM evaluation on these questions is presented in Section 4.3. The experimental setup is presented in section 4.4. The results of experiment runs are discussed in Section 4.4. Limitations and steps taken to overcome them are described in Section ???. The chapter then ends with the conclusion in Section 4.6 and thoughts on future work in Section ???.

4.2 Preliminaries

Scenario reasoning is a subset of reasoning types that are relevant to scenario-based testing of AVs. The term reasoning can be defined as a sequence of intermediate statements written by the LLM beginning with a set of statements and ending with a conclusive statement. There are several types of reasoning based on the nature of this sequence such as deductive and inductive reasoning. We need to prioritize the reasoning types that are relevant to the topic of scenario based testing of AVs. This prioritization process considers the representations of a scenario considered in Chapter 2 such as the *6 layer model*. Here, due to the spatial and temporal layers (layer 1-3 and layer 4) model, spatial and temporal reasoning types are included in scenario reasoning. Due to the layers in this model that

refer to vehicles, pedestrians and roads, we also include commonsense reasoning along with deductive and inductive reasoning.

As stated in Section 4.1, since the possible LLM inputs and outputs are *open*, it is important to carefully design and experiment with prompts such that the outputs *align* with the user intent [69]. The subfield of *prompt engineering* covers approaches to this end [119]. In CruzwayQA, we consider prominent prompt engineering approaches from the literature such as *chain-of-thought (CoT)* prompting [146].

4.3 Method

CruzwayQA is a concepts-based, synthetic QA data generation approach to measure scenario reasoning capabilities in LLMs. We use LLMs to generate synthetic scenarios in the narrative form and generate multiple-choice questions (MCQs) based on that. Since we are only using LLMs to generate the MCQ strings that adhere with the structures described in the prompt, there is a significant prompt engineering process involved. This results in a *modular prompt* and consists of the ontologies, standard reasoning styles (which constitute scenario reasoning as described in section 4.2), MCQ formats and concepts from the Bloom’s taxonomy to enhance questions. A modular prompt separates ontologies and domain-agnostic QA structures thereby giving flexibility in reuse in other domains. An overview of the QA generation prompt design is presented in Figure 4.2.

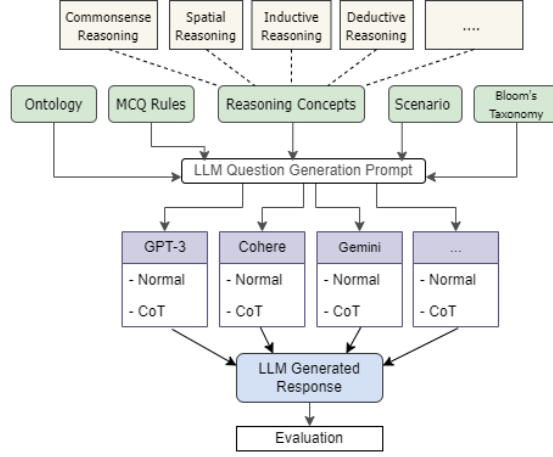


Figure 4.2: ScenarioQA Overview

4.3.1 Scenario Ontology

The *OpenXOntology* is a standardized organization of concepts relevant to scenario-based testing of autonomous vehicles created by the ASAM organization [10]. This ontology consists of three key subontologies: core, domain and application. The core ontology contains *meta* information; this would be concepts about the ontology itself, such as the types of relations. The domain ontology is the one relevant for our purposes. It contains concepts detailed in the six layer model of a scenario [121], broadly the static and dynamic elements of a scenario. The application ontology contains concepts that can be modified for other standards such as ASAM OpenDRIVE for roads and ASAM OpenSCENARIO for dynamic aspects of a scenario. In CruzwayQA, we consider a *general subset* of this ontology such as concepts about each layer of the 6 layer model (broadly roads, vehicles and pedestrians). We also consider *refining* a given subset of this ontology as presented by Tang et al. [136]. This implies adding more concepts and making new relations among these concepts. Similar to this work, to represent the ontology in the prompt, we make use of the *digraph* format.

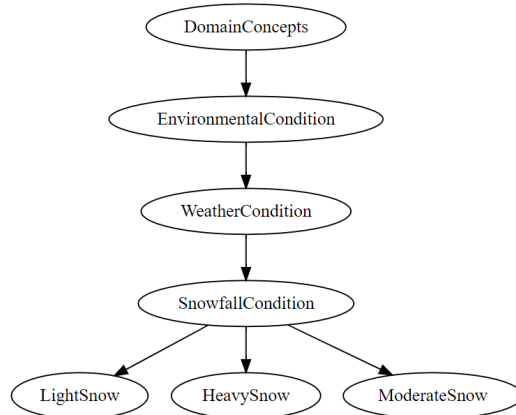


Figure 4.3: An example ontology from the ASAM OpenXOntology standard

You are a road safety scientist at a vehicle company with a background in traffic engineering, civil engineering and autonomous vehicles. I have a road driving scenario ontology as shown below (in DOT format).

```

digraph Ontology {
DrivingActivity -> ChangeLane;
DrivingActivity -> CutIn;
DrivingActivity -> Overtake;
DrivingActivity -> MakeATurn;
HumanActivity -> [PedestrianActivity, DriverActivity, RiderActivity];
MakeATurn -> [MakeALeftTurn, MakeARightTurn];
MovingActivity -> [Accelerate, Decelerate, Stop, PedestrianActivity,
    Turn, Cross];
}

```

Listing 4.1: *Initial ontology provided to the LLM for refining. The ontology refinement prompt is appended to this. Modifications for this are also considered such as one without pedestrian activity or turning activity.*

Add 5 new relevant concepts, terms or entities to the ontology considering the following instructions:

- * Each class has only one parent class.

```

* You should merge duplicated concepts.
* You should delete concepts irrelevant to road driving scenarios.
* You should improve the ontology structure.
* Output the new ontology in DOT format.
* The parent class and child class should be arranged as: parent
    class -> child class.
* Please prune the tree to only contain important concepts.
* Please increase tree depth.
* Please ensure that the concepts generated are consistent and
    plausible with respect to the previously generated concepts.
* Please add some interesting concepts.
* Please do not write anything else other than the concepts.

```

Listing 4.2: *Prompt given to the LLM to refine a given ontology and add 10 new concepts. The ontology is generated in the Graphviz digraph format to represent the tree structure.*

Reasoning Types

We select certain reasoning categories mentioned in [16] that are relevant to scenario based testing of AVs such as commonsense reasoning, deductive and inductive reasoning, spatial reasoning and temporal reasoning. For commonsense reasoning questions, we consider concepts and relations from CommonsenseQA, which considers concepts from ConceptNet [134]. Some of these relations are *cause-effect*, *at-location*, *has-property*. Deductive reasoning involves making inferences from statements and assumptions. We consider categories of inferences from EntailmentBank [29]. For spatial reasoning, we consider relations presented in *SpartQA* [96]. These include relative directions such as *left* and *right* and distances such as *near* and *far*. For temporal reasoning, we make use of relations in *Timedial*. We make use of concepts for frequency, ordering (such as before or

later) and duration.

Instead of explicitly describing these reasoning styles in the prompt, we provide them as CoT examples as presented in the work by Wei et al. [146]. Consistent with in-context learning capabilities in LLMs, this ensures that the LLM implicitly learns the patterns in these examples. We consider examples of these reasoning styles in the works above, write similar examples or separately have an LLM generate similar examples and add them to the prompt after verification.

Along with MCQ examples in accordance with the reasoning types stated above, we also include other the question formation structures described in the subsections ahead. The final result is a prompt based on which an LLM generates a set of MCQs.

Multiple Choice Question Structure

We consider the multiple choice question (MCQ) structure from [36]. Here, an MCQ is defined as a question where multiple options for answers are arranged with one being the correct answer called the *stem* and others being the *distractors*. Similar to [36], the questions generally are assigned a category by making use of 6 action words Bloom’s taxonomy.

Here is some additional information about multiple-choice question formation as well as Bloom’s taxonomy:

A multiple-choice question is a collection of three components aimed at testing a students understanding of a certain topic, given a particular context of what the student is expected to know. The topic, as well as the context of the topic, will be provided in order to generate effective multiple-choice questions. The three components of a multiple-choice question are as follows: a Stem, a Correct Answer, and two Distractors. There must always be only one correct answer and only two distractors.

The stem refers to the question the student will attempt to answer, as well as the relevant context necessary in order to answer the question. It may be in the form of a question, an incomplete statement, or a scenario. The stem should focus on assessing the specific knowledge or concept the question aims to evaluate.

The Correct Answer refers to the correct, undisputable answer to the question in the stem.

A Distractor is an incorrect answer to the question in the stem and adheres to the following properties. [] Use ‘‘None of the Above’’ or ‘‘All of the Above’’ style answer choices sparingly. These answer choices have been shown to, in general, be less effective at measuring or assessing student understanding.

Multiple-choice questions should be clear, concise, and grammatically correct statements. Make sure the questions are worded in a way that is easy to understand and does not introduce unnecessary complexity or ambiguity. Students should be able to understand the questions without confusion. The question should not be too long, and allow most students to finish in less than five minutes. This means adhering to the following properties.

Blooms’ Taxonomy and Action Verbs:

Multiple-choice questions must be well aligned to the learning objectives they are intended to assess students knowledge on. This implies that they must assess skills at the right cognitive level corresponding to the Bloom’s taxonomy categorization of the learning objective. Bloom’s Taxonomy offers a framework for categorizing the depth of learning, and it provides guidance on selecting appropriate action verbs when writing learning objectives. Here are the six levels of Bloom’s taxonomy and their definitions:

Remember – This level involves retrieving , recognizing , and recalling relevant knowledge from long–term memory.

Understand – At this level , learners construct meaning from oral , written , and graphic messages through interpreting , exemplifying , classifying , summarizing , inferring , comparing , and explaining .

Apply – This level requires learners to carry out or use a procedure through executing or implementing it .

Analyze – At this level , learners break material into constituent parts , determine how the parts relate to one another and to an overall structure or purpose through differentiating , organizing .

Evaluate – This level involves making judgments based on criteria and standards through checking and critiquing .

Create – At this level , learners put elements together to form a coherent or functional whole , or they reorganize elements into a new pattern or structure through generating .

Given information about the traffic related ontology , as well as question formation guidelines and Bloom’s taxonomy , would you be able to create multiple–choice question for one TYPE OF INFERENCE provided below that fulfill the MCQ guidelines as well as Bloom’s taxonomy? The below example shows the type of inference relation as well as an example of the types of questions asked with that particular relation . Please pick one type of inference and one Bloom’s taxonomy term at random . For option D, make sure the option is similar to the answer to the question , however incorrect .

Relation

AtLocation , Causes , CapableOf , Antonym , HasPrerequisite , CausesDesire
Desires , PartOf , HasProperty

Formulated question example corresponding to the relations above:

Where would you least want an autonomous vehicle's pedestrian detection system to fail?

- A. Downtown crosswalk
- B. Closed proving grounds
- C. Virtual simulation lab
- D. Maintenance garage

What is the primary benefit of calibrating an autonomous vehicle's LIDAR sensors before on-road testing?

- A. More accurate distance measurements
- B. Reduced sensor power consumption
- C. Increased false positives
- D. Slower scanning rates

Why might engineers increase the resolution of the forward-facing camera on an autonomous vehicle?

- A. To detect small objects at greater distances
- B. To reduce the size of recorded image files
- C. To simplify the image-processing pipeline
- D. To limit the sensors frame rate

If you want to evaluate sensor performance in conditions opposite to clear daylight, which scenario would you choose?

- A. Nighttime driving
- B. Midday under bright sun
- C. Late-morning with clear skies
- D. Early-afternoon sunlight

Before running a lane-keeping assist test, what must engineers do first?

- A. Calibrate steering and inertial sensors
- B. Increase the vehicles top speed

- C. Disable all safety overrides
- D. Update the onboard software

Repeated phantom braking events in road trials typically cause engineers to do what?

- A. Refine the braking algorithm
- B. Lower the test speed limit
- C. Turn off automatic emergency braking
- D. Ignore the anomalies

What outcome do safety engineers most desire when validating an autonomous vehicle on public roads?

- A. Consistent system reliability
- B. Maximum travel speed
- C. Minimal travel time
- D. Lowest energy consumption

Which component is part of an autonomous vehicles perception stack?

- A. LIDAR sensor
- B. Fuel injector
- C. Exhaust manifold
- D. Transmission clutch

What property should a closed-course test scenario have to ensure consistent results across multiple runs?

- A. Repeatability
- B. Randomness
- C. Ambiguity
- D. Variability

Now we will generate the questions. Please remember that any of the options A or B or C or D can be the correct one and ordering can be an important aspect.

Please shuffle the ordering of the option that is correct. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.

A note about the difficulty of questions: The difficulty must be quite significant in terms of the reasoning needed to answer the questions. It should be quite challenging to arrive at conclusions given the premises of

the questions. In addition the alternatives should be confusing.

Please format the question and answer data in the form of a dictionary as follows:

<open curly bracket>

"Scenario": "<A scenario description in narrative form based on a subset of the ontology concepts mentioned above. Several questions will be written on the basis of this scenario, so please present the the necessary details accurately. Please describe the scenario as a sequence of events that are unfolding. You do not need to fully cover the scenario concepts that are mentioned above, only a subset of it. Please ensure that the scenario is plausible.>",

"Questions": <open curly bracket>

"<question number label written as Q1, Q2 and so on>":<open curly bracket>

"Relation": "<please mention the relation selected from above. Please select only one of the terms.>",

"Bloom": "<please mention the Bloom's taxonomy term selected from above. Please select only one of the terms.>"

```

"Question": "A very challenging question based on a synthesis of the
    scenario , relation type , Bloom's taxonomy term and implicitly
    similar in structure to the examples mentioned above with the
    intent to evaluate a large language model for reasoning
    effectively . Most question contents should be in the scenario
    information shared . As a result , please keep the questions
    concise." ,
"A": "<Details of option A.>" ,
"B": "<Details of option B. Since it is an MCQ type evaluation , we
    must make sure that the question isn't already answered in other
    options or in other questions.>" ,
"C": "<Details of option C. Since it is an MCQ type evaluation , we
    must make sure that the question isn't already answered in other
    options or in other questions.>" ,
"D": "<Details of option D. Since it is an MCQ type evaluation , we
    must make sure that the question isn't already answered in other
    options or in other questions.>" ,
"Answer rationale": "<string describing answer rationale>" ,
"Correct option": "<A or B or C or D that is the correct option. This
    will be used as the reference , so please make sure it is
    absolutely correct in accordance with the rationale written above
    .>"
<close curly bracket (specific question)>
... ,
...
<close curly bracket (all questions)>
<close curly bracket>

```

Please only generate the json dictionary exactly as specified . Please don't generate anything else .

```
Nothing else. Nothing else. Nothing else. Please do not add any tags  
of any sort, just the formatting shown here.
```

Listing 4.3: QA structure prompt for the commonsense reasoning QA data generation task. The examples are generated after several iterations with another LLM and are based on the respective works cited in Section 4.3.1

```
Here is some additional information about multiple-choice question  
formation as well as Bloom's taxonomy:
```

```
A multiple-choice question is a collection of three components aimed  
at testing a student's understanding of a  
certain topic, given a particular context of what the student is  
expected to know. The topic, as well as the context of the topic,  
will be provided in order to generate effective multiple-choice  
questions. The three components of a multiple-choice question are  
as follows: a Stem, a Correct Answer, and two Distractors. There  
must always be only one correct answer and only two distractors.
```

```
The stem refers to the question the student will attempt to answer,  
as well as the relevant context necessary in order to answer the  
question. It may be in the form of a question, an incomplete  
statement, or a scenario. The stem should focus on assessing the  
specific knowledge or concept the question aims to evaluate.
```

```
The Correct Answer refers to the correct, undisputable answer to the  
question in the stem.
```

```
A Distractor is an incorrect answer to the question in the stem and  
adheres to the following properties. Use "None of the Above" or  
"All of the Above" style answer choices sparingly. These  
answer choices have been shown to, in general, be less effective  
at measuring or assessing student understanding.
```

```
Multiple-choice questions should be clear, concise, and grammatically
```

correct statements. Make sure the questions are worded in a way that is easy to understand and does not introduce unnecessary complexity or ambiguity. Students should be able to understand the questions without confusion. The question should not be too long, and allow most students to finish in less than five minutes. This means adhering to the following properties.

Blooms' Taxonomy and Action Verbs:

Multiple-choice questions must be well aligned to the learning objectives they are intended to assess students' knowledge on. This implies that they must assess skills at the right cognitive level corresponding to the Bloom's taxonomy categorization of the learning objective. Bloom's Taxonomy offers a framework for categorizing the depth of learning, and it provides guidance on selecting appropriate action verbs when writing learning objectives. Here are the six levels of Bloom's taxonomy and their definitions:

Remember – This level involves retrieving, recognizing, and recalling relevant knowledge from long-term memory.

Understand – At this level, learners construct meaning from oral, written, and graphic messages through interpreting, exemplifying, classifying, summarizing, inferring, comparing, and explaining.

Apply – This level requires learners to carry out or use a procedure through executing or implementing it.

Analyze – At this level, learners break material into constituent parts, determine how the parts relate to one another and to an overall structure or purpose through differentiating, organizing.

Evaluate – This level involves making judgments based on criteria and standards through checking and critiquing.

Create – At this level, learners put elements together to form a coherent or functional whole, or they reorganize elements into a

new pattern or structure through generating.

Given information about the traffic related ontology, as well as question formation guidelines and Bloom's taxonomy, would you be able to create multiple-choice questions for EACH TYPE OF INFERENCE provided below that fulfill the MCQ guidelines as well as Bloom's taxonomy? The below example shows the type of inference relation as well as an example of the types of questions asked with that particular relation. Create one question for each type of relation. For option D, make sure the option is similar to the answer to the question, however incorrect. Also, create an answer sheet with the answers to the questions.

Relation

AtLocation, Causes, CapableOf, Antonym, HasPrerequisite, CausesDesire
Desires, PartOf, HasProperty

Formulated question example corresponding to the relations above:

Where would you least want an autonomous vehicle's pedestrian detection system to fail?

- A. Downtown crosswalk
- B. Closed proving grounds
- C. Virtual simulation lab
- D. Maintenance garage

What is the primary benefit of calibrating an autonomous vehicle's LIDAR sensors before on-road testing?

- A. More accurate distance measurements
- B. Reduced sensor power consumption
- C. Increased false positives
- D. Slower scanning rates

Why might engineers increase the resolution of the forward-facing camera on an autonomous vehicle?

- A. To detect small objects at greater distances
- B. To reduce the size of recorded image files
- C. To simplify the image-processing pipeline
- D. To limit the sensor's frame rate

If you want to evaluate sensor performance in conditions opposite to clear daylight, which scenario would you choose?

- A. Nighttime driving
- B. Midday under bright sun
- C. Late-morning with clear skies
- D. Early-afternoon sunlight

Before running a lane-keeping assist test, what must engineers do first?

- A. Calibrate steering and inertial sensors
- B. Increase the vehicle's top speed
- C. Disable all safety overrides
- D. Update the onboard software

Repeated phantom braking events in road trials typically cause engineers to do what?

- A. Refine the braking algorithm
- B. Lower the test speed limit
- C. Turn off automatic emergency braking
- D. Ignore the anomalies

What outcome do safety engineers most desire when validating an autonomous vehicle on public roads?

- A. Consistent system reliability

- B. Maximum travel speed
- C. Minimal travel time
- D. Lowest energy consumption

Which component is part of an autonomous vehicle's perception stack?

- A. LIDAR sensor
- B. Fuel injector
- C. Exhaust manifold
- D. Transmission clutch

What property should a closed-course test scenario have to ensure consistent results across multiple runs?

- A. Repeatability
- B. Randomness
- C. Ambiguity
- D. Variability

Now we will generate the questions. Please remember that any of the options A or B or C or D can be the correct one and ordering can be an important aspect.

Please shuffle the ordering of the option that is correct. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.

A note about the difficulty of questions: The difficulty must be quite significant in terms of the reasoning needed to answer the questions. It should be quite challenging to arrive at conclusions given the premises of the questions. In addition the alternatives should be confusing.

Please format the question and answer data in the form of a dictionary as follows:

<open curly bracket>

"Scenario": "<A scenario description in narrative form based on a subset of the ontology concepts mentioned above. Several questions will be written on the basis of this scenario, so please present the the necessary details accurately. Please describe the scenario as a sequence of events that are unfolding. You do not need to fully cover the scenario concepts that are mentioned above, only a subset of it. Please ensure that the scenario is plausible.>",

"Questions": <open curly bracket>

"<question number label written as Q1, Q2 and so on>":<open curly bracket>

"Relation": "<please mention the relation selected from above. Please select only one of the terms.>",

"Bloom": "<please mention the Bloom's taxonomy term selected from above. Please select only one of the terms.>"

"Question": "A very challenging question based on a synthesis of the scenario, relation type, Bloom's taxonomy term and implicitly similar in structure to the examples mentioned above with the intent to evaluate a large language model for reasoning effectively. Most question contents should be in the scenario information shared. As a result, please keep the questions concise.",

"A": "<Details of option A.>",

"B": "<Details of option B. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.>",

"C": "<Details of option C. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.>",

"D": "<Details of option D. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other

```

    options or in other questions.>",
"Answer rationale": "<string describing answer rationale>",
"Correct option": "<A or B or C or D that is the correct option. This
    will be used as the reference , so please make sure it is
    absolutely correct in accordance with the rationale written above
    .>"
<close curly bracket (specific question)>
... ,
...
<close curly bracket (all questions)>
<close curly bracket>

Please only generate the json dictionary exactly as specified. Please
    don't generate anything else.
Nothing else. Nothing else. Nothing else. Please do not add any tags
    of any sort , just the formatting shown here.

```

Listing 4.4: *QA structure prompt for the Commonsense Reasoning QA data generation task. The examples are generated after several iterations with another LLM and are based on the respective works cited in Section 4.3.1*

Here is some additional information about multiple-choice question formation as well as Bloom's taxonomy:

A multiple-choice question is a collection of three components aimed at testing a student's understanding of a certain topic, given a particular context of what the student is expected to know. The topic, as well as the context of the topic, will be provided in order to generate effective multiple-choice questions. The three components of a multiple-choice question are as follows: a Stem, a Correct Answer, and two Distractors. There must always be only one correct answer and only two distractors.

The stem refers to the question the student will attempt to answer, as well as the relevant context necessary in order to answer the question. It may be in the form of a question, an incomplete statement, or a scenario. The stem should focus on assessing the specific knowledge or concept the question aims to evaluate.

The Correct Answer refers to the correct, undisputable answer to the question in the stem.

A Distractor is an incorrect answer to the question in the stem and adheres to the following properties. Use “None of the Above” or “All of the Above” style answer choices sparingly. These answer choices have been shown to, in general, be less effective at measuring or assessing student understanding.

Multiple-choice questions should be clear, concise, and grammatically correct statements. Make sure the questions are worded in a way that is easy to understand and does not introduce unnecessary complexity or ambiguity. Students should be able to understand the questions without confusion. The question should not be too long, and allow most students to finish in less than five minutes. This means adhering to the following properties.

Blooms’ Taxonomy and Action Verbs:

Multiple-choice questions must be well aligned to the learning objectives they are intended to assess students’ knowledge on. This implies that they must assess skills at the right cognitive level corresponding to the Bloom’s taxonomy categorization of the learning objective. Bloom’s Taxonomy offers a framework for categorizing the depth of learning, and it provides guidance on

selecting appropriate action verbs when writing learning objectives. Here are the six levels of Bloom's taxonomy and their definitions:

Remember — This level involves retrieving, recognizing, and recalling relevant knowledge from long-term memory.

Understand — At this level, learners construct meaning from oral, written, and graphic messages through interpreting, exemplifying, classifying, summarizing, inferring, comparing, and explaining.

Apply — This level requires learners to carry out or use a procedure through executing or implementing it.

Analyze — At this level, learners break material into constituent parts, determine how the parts relate to one another and to an overall structure or purpose through differentiating, organizing.

Evaluate — This level involves making judgments based on criteria and standards through checking and critiquing.

Create — At this level, learners put elements together to form a coherent or functional whole, or they reorganize elements into a new pattern or structure through generating.

Given information about the traffic related ontology, as well as question formation guidelines and Bloom's taxonomy, would you be able to create multiple-choice questions for EACH TYPE OF INFERENCE provided below that fulfill the MCQ guidelines as well as Bloom's taxonomy? The below example shows the type of inference relation as well as an example of the types of questions asked with that particular relation. Create one question for each type of relation. For option D, make sure the option is similar to the answer to the question, however incorrect. Also, create an answer sheet with the answers to the

questions.

Structure: Inference Type $\rightarrow$ Infering a conclusion from Fact 1 and Fact 2

Substitution:

Fact 1: when a light wave hits a reflective object, the light wave will be reflected

Fact 2: a mirror is a kind of reflective object

Conclusion: when a light wave hits a mirror, the light wave will be reflected

Inference from Rule:

Fact 1: if two species have similar characteristics, they may share a common ancestor

Fact 2: rhinoceroses and horses have similar characteristics

Conclusion: rhinoceroses and horses might share a common ancestor

Further Specificatio or Conjunction:

Fact 1: an animal requires warmth for survival as the season changes to winter

Fact 2: thick fur can be used for keeping warm

Conclusion: thick fur can be used for keeping warm as the season changes to winter

Infer Class from Properties:

Fact 1: A compound is made of two or more elements chemically combined

Fact 2: sodium chloride is made of two elements chemically combined

Conclusion: sodium chloride is a kind of compound

Property Inheritance:

Fact 1: an animal's shell is usually hard

Fact 2: something hard can be used for protection

Conclusion: an animal's shell is usually hard for protection

Sequential Inference:

Fact 1: In molecular biology, translation follows transcription

Fact 2: transcription is when genetic information flows from DNA to RNA translation is when genetic information flows from RNA to proteins

Conclusion: In molecular biology, genetic information flows from DNA to RNA to proteins

Now we will generate the questions. Please remember that any of the options A or B or C or D can be the correct one and ordering can be an important aspect.

Please shuffle the ordering of the option that is correct. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.

Please format the question and answer data in the form of a dictionary as follows:

<open curly bracket>

"Scenario": "<A scenario description in narrative form based on a subset of the ontology concepts mentioned above. Several questions will be written on the basis of this scenario, so please present the the necessary details accurately. Please describe the scenario as a sequence of events that are unfolding. You do not need to fully cover the scenario concepts that are mentioned above, only a subset of it. Please ensure that the scenario is plausible.>",

"Questions": <open curly bracket>

"<question number label written as Q1, Q2 and so on>":<open curly bracket>

```

"Relation": "<please mention the relation selected from above. Please
    select only one of the terms.>",
"Bloom": "<please mention the Bloom's taxonomy term selected from
    above. Please select only one of the terms.>"
"Question": "A very challenging question based on a synthesis of the
    scenario, relation type, Bloom's taxonomy term and implicitly
    similar in structure to the examples mentioned above with the
    intent to evaluate a large language model for reasoning
    effectively. Most question contents should be in the scenario
    information shared. As a result, please keep the questions
    concise.",
"A": "<Details of option A.>",
"B": "<Details of option B. Since it is an MCQ type evaluation, we
    must make sure that the question isn't already answered in other
    options or in other questions.>",
"C": "<Details of option C. Since it is an MCQ type evaluation, we
    must make sure that the question isn't already answered in other
    options or in other questions.>",
"D": "<Details of option D. Since it is an MCQ type evaluation, we
    must make sure that the question isn't already answered in other
    options or in other questions.>",
"Answer rationale": "<string describing answer rationale>",
"Correct option": "<A or B or C or D that is the correct option. This
    will be used as the reference, so please make sure it is
    absolutely correct in accordance with the rationale written above
    .>"
<close curly bracket (specific question)>
...
...
<close curly bracket (all questions)>
<close curly bracket>

```

```
Please only generate the json dictionary exactly as specified. Please
    don't generate anything else.
Nothing else. Nothing else. Nothing else. Please do not add any tags
    of any sort, just the formatting shown here.
```

Listing 4.5: *QA structure prompt for the Deductive Reasoning QA data generation task. The examples are generated after several iterations with another LLM and are based on the respective works cited in Section 4.3.1*

Here is some additional information about multiple-choice question formation as well as Bloom's taxonomy:

A multiple-choice question is a collection of three components aimed at testing a student's understanding of a certain topic, given a particular context of what the student is expected to know. The topic, as well as the context

of the topic, will be provided in order to generate effective multiple-choice questions. The three components of a multiple-choice question are as follows: a Stem, a Correct Answer, and two Distractors. There must always be only one correct answer and only two distractors.

The stem refers to the question the student will attempt to answer, as well as the relevant context necessary in order to answer the question. It may be in the form of a question, an incomplete statement, or a scenario. The stem should focus on assessing the specific knowledge or concept the question aims to evaluate.

The Correct Answer refers to the correct, undisputable answer to the question in the stem.

A Distractor is an incorrect answer to the question in the stem and

adheres to the following properties. Use “None of the Above” or “All of the Above” style answer choices sparingly. These answer choices have been shown to, in general, be less effective at measuring or assessing student understanding.

Multiple-choice questions should be clear, concise, and grammatically correct statements. Make sure the questions are worded in a way that is easy to understand and does not introduce unnecessary complexity or ambiguity. Students should be able to understand the questions without confusion. The question should not be too long, and allow most students to finish in less than five minutes. This means adhering to the following properties.

Blooms’ Taxonomy and Action Verbs:

Multiple-choice questions must be well aligned to the learning objectives they are intended to assess students’ knowledge on. This implies that they must assess skills at the right cognitive level corresponding to the Bloom’s taxonomy categorization of the learning objective. Bloom’s Taxonomy offers a framework for categorizing the depth of learning, and it provides guidance on selecting appropriate action verbs when writing learning objectives. Here are the six levels of Blooms taxonomy and their definitions:

Remember – This level involves retrieving, recognizing, and recalling relevant knowledge from long-term memory.

Understand – At this level, learners construct meaning from oral, written, and graphic messages through interpreting, exemplifying, classifying, summarizing, inferring, comparing, and explaining.

Apply – This level requires learners to carry out or use a procedure through executing or implementing it.

Analyze – At this level, learners break material into constituent

parts, determine how the parts relate to one another and to an overall structure or purpose through differentiating, organizing. Evaluate – This level involves making judgments based on criteria and standards through checking and critiquing. Create – At this level, learners put elements together to form a coherent or functional whole, or they reorganize elements into a new pattern or structure through generating.

Given information about the traffic related ontology, as well as question formation guidelines and Bloom's taxonomy, would you be able to create multiple-choice questions for EACH TYPE OF INFERENCE provided below that fulfill the MCQ guidelines as well as Bloom's taxonomy? The below example shows the type of inference relation as well as an example of the types of questions asked with that particular relation. Create one question for each type of relation. For option D, make sure the option is similar to the answer to the question, however incorrect. Also, create an answer sheet with the answers to the questions.

Here are some examples. I want you to consider them, adapt the structure within these examples.

Leader Follower Inference

Statement Given:

In a highway-platooning test, Vehicle C3 trailed Vehicle T8 at a fixed 20 m gap.

During that same test, Vehicle L5 served as the designated platoon leader for Vehicle T8.

Question: Vehicle L5 is the ____ of Vehicle C3.

A. follower

B. neighbor

C. leader

D. observer

Answer: C. leader

Sibling-and-Parent Child Inference

Statement Given:

Vehicles R2 and N7 were both platoon followers directly behind lead
Vehicle P1.

Vehicle R2 received its speed commands from the central controller CC
.

Question: Vehicle N7 is the ____ of the central controller CC.

A. follower

B. leader

C. peer

D. observer

Answer: A. follower

Parent-of-One-Sibling Parent-of-Another Inference

Statement Given:

In a convoy scenario, Vehicle Lead1 acted as platoon leader for
Vehicle FollowA.

Vehicle FollowA drove side-by-side (lane-adjacent) with Vehicle SideB
.

Question: Vehicle Lead1 is the ____ of Vehicle SideB.

A. neighbor

B. controller

C. leader

D. observer

Answer: C. leader

Shared-Parent Sibling (Neighbor) Inference

Statement Given:

Vehicles P4 and Q6 both followed Vehicle LeaderX in adjacent lanes on the highway.

Vehicle P4 braked suddenly to avoid a stopped obstacle.

Question: Vehicle Q6 is the ____ of Vehicle P4.

- A. leader
- B. neighbor
- C. controller
- D. follower

Answer: B. neighbor

Grandparent-Style (Overarching) Scenario Inference

Statement Given:

The Highway Driving Scenario (HDS) encompasses a Lane-Change Sub-Scenario (LCS).

The Lane-Change Sub-Scenario (LCS) further defines the Cut-In Event Sub-Scenario (CIES).

Question: HDS is the ____ of CIES.

- A. sub-scenario
- B. overarching scenario
- C. peer scenario
- D. nested scenario

Answer: B. overarching scenario

Nephew-Style (Indirect Sub-Scenario) Inference

Statement Given:

Urban Driving Scenario (UDS) and Suburban Driving Scenario (SDS) are parallel sub-scenarios under the General Driving Test Suite (GDTS).

UDS further includes a Parking-Maneuver Sub-Scenario (PMS).

Question: PMS is an ____ with respect to SDS.

- A. direct sub-scenario
- B. indirect sub-scenario

C. peer scenario

D. overarching scenario

Answer: B. indirect sub-scenario

Now we will generate the questions. Please remember that any of the options A or B or C or D can be the correct one and ordering can be an important aspect.

Please shuffle the ordering of the option that is correct. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.

Please format the question and answer data in the form of a dictionary as follows:

<open curly bracket>

"Scenario": "<A scenario description in narrative form based on a subset of the ontology concepts mentioned above. Several questions will be written on the basis of this scenario, so please present the the necessary details accurately. Please describe the scenario as a sequence of events that are unfolding. You do not need to fully cover the scenario concepts that are mentioned above, only a subset of it. Please ensure that the scenario is plausible.>",

"Questions": <open curly bracket>

"<question number label written as Q1, Q2 and so on>":<open curly bracket>

"Relation": "<please mention the relation selected from above. Please select only one of the terms.>",

"Bloom": "<please mention the Bloom's taxonomy term selected from above. Please select only one of the terms.>"

"Question": "A very challenging question based on a synthesis of the scenario, relation type, Bloom's taxonomy term and implicitly

```

    similar in structure to the examples mentioned above with the
    intent to evaluate a large language model for reasoning
    effectively. Most question contents should be in the scenario
    information shared. As a result, please keep the questions
    concise.",
"A": "<Details of option A.>",
"B": "<Details of option B. Since it is an MCQ type evaluation, we
    must make sure that the question isn't already answered in other
    options or in other questions.>",
"C": "<Details of option C. Since it is an MCQ type evaluation, we
    must make sure that the question isn't already answered in other
    options or in other questions.>",
"D": "<Details of option D. Since it is an MCQ type evaluation, we
    must make sure that the question isn't already answered in other
    options or in other questions.>",
"Answer rationale": "<string describing answer rationale>",
"Correct option": "<A or B or C or D that is the correct option. This
    will be used as the reference, so please make sure it is
    absolutely correct in accordance with the rationale written above
    .>"
<close curly bracket (specific question)>
...
...
<close curly bracket (all questions)>
<close curly bracket>

Please only generate the json dictionary exactly as specified. Please
    don't generate anything else.
Nothing else. Nothing else. Nothing else. Please do not add any tags

```

of any sort , just the formatting

Listing 4.6: *QA structure prompt for the inductive reasoning QA data generation task. The examples are generated after several iterations with another LLM and are based on the respective works cited in Section 4.3.1*

Here is some additional information about multiple-choice question formation as well as Bloom’s taxonomy:

A multiple-choice question is a collection of three components aimed at testing a student’s understanding of a certain topic , given a particular context of what the student is expected to know. The topic , as well as the context of the topic , will be provided in order to generate effective multiple-choice questions. The three components of a multiple-choice question are as follows: a Stem, a Correct Answer, and two Distractors. There must always be only one correct answer and only two distractors.

The stem refers to the question the student will attempt to answer , as well as the relevant context necessary in order to answer the question. It may be in the form of a question , an incomplete statement , or a scenario. The stem should focus on assessing the specific knowledge or concept the question aims to evaluate.

The Correct Answer refers to the correct , undisputable answer to the question in the stem.

A Distractor is an incorrect answer to the question in the stem and adheres to the following properties. Use “None of the Above” or “All of the Above” style answer choices sparingly. These answer choices have been shown to , in general , be less effective at measuring or assessing student understanding.

Multiple-choice questions should be clear, concise, and grammatically correct statements. Make sure the questions are worded in a way that is easy to understand and does not introduce unnecessary complexity or ambiguity. Students should be able to understand the questions without confusion. The question should not be too long, and allow most students to finish in less than five minutes. This means adhering to the following properties.

Blooms' Taxonomy and Action Verbs:

Multiple-choice questions must be well aligned to the learning objectives they are intended to assess students' knowledge on. This implies that they must assess skills at the right cognitive level corresponding to the Bloom's taxonomy categorization of the learning objective. Bloom's Taxonomy offers a framework for categorizing the depth of learning, and it provides guidance on selecting appropriate action verbs when writing learning objectives. Here are the six levels of Bloom's taxonomy and their definitions:

Remember – This level involves retrieving, recognizing, and recalling relevant knowledge from long-term memory.

Understand – At this level, learners construct meaning from oral, written, and graphic messages through interpreting, exemplifying, classifying, summarizing, inferring, comparing, and explaining. Apply – This level requires learners to carry out or use a procedure through executing or implementing it.

Analyze – At this level, learners break material into constituent parts, determine how the parts relate to one another and to an overall structure or purpose through differentiating, organizing. Evaluate – This level involves making judgments based on criteria and standards through checking and critiquing.

Create – At this level, learners put elements together to form a coherent or functional whole, or they reorganize elements into a new pattern or structure through generating.

Given information about the traffic related ontology, as well as question formation guidelines and Bloom's taxonomy, would you be able to create multiple-choice questions for EACH TYPE OF INFERENCE provided below that fulfill the MCQ

guidelines as well as Bloom's taxonomy? The below example shows the type of inference relation as well as an example of the types of questions asked with that particular relation. Create one question for each type of relation. For option D, make sure the option is similar to the answer to the question, however incorrect. Also, create an answer sheet with the answers to the questions.

Here are some examples. I want you to consider them, adapt the structure within these examples.

Typical spatial reasoning relations that can be drawn based on the example below

Spatial relations: Relative positions.

Example 1

Question: My autonomous car is stopped at a traffic light. From a birds-eye view of my vehicle, a delivery van is in the top-left quadrant. A pedestrian then appears directly south of the van (toward my car). What is the pedestrian's position relative to my vehicle?

Options:

(incorrect) Top-left

(incorrect) Top-right

(correct) Bottom-left

(incorrect) Bottom-right

Example 2

Question: My self-driving shuttle is idling on the curb. On the birds-eye view, an electric scooter stands in the bottom-left quadrant. A charging robot is placed directly north of the scooter (away from my shuttle). Where is the robot relative to my vehicle?

Options:

(incorrect) Bottom-left

(incorrect) Bottom-right

(correct) Top-left

(incorrect) Top-right

Example 3

Question: My automated taxi is waiting for passengers. From the top-down view, a bicycle is parked in the top-right quadrant relative to my taxi. A cone barrier is set directly south of that bicycle (toward my taxi). What quadrant is the barrier in relative to my taxi?

Options:

(incorrect) Top-right

(incorrect) Top-left

(correct) Bottom-right

(incorrect) Bottom-left

Example 4

Question: My autonomous car is docked in a parking bay. A van occupies the top-left quadrant in a birds-eye view. A manhole cover is located directly east of this van. What is the manhole covers position relative to my vehicle?

Options:

(incorrect) Top-left
(incorrect) Bottom-left
(incorrect) Bottom-right
(correct) Top-right

Now we will generate the questions. Please remember that any of the options A or B or C or D can be the correct one and ordering can be an important aspect.

Please shuffle the ordering of the option that is correct. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.

Please format the question and answer data in the form of a dictionary as follows:

<open curly bracket>

"Scenario": "<A scenario description in narrative form based on a subset of the ontology concepts mentioned above. Several questions will be written on the basis of this scenario, so please present the the necessary details accurately. Please describe the scenario as a sequence of events that are unfolding. You do not need to fully cover the scenario concepts that are mentioned above, only a subset of it. Please ensure that the scenario is plausible.>",

"Questions": <open curly bracket>

"<question number label written as Q1, Q2 and so on>":<open curly bracket>

"Relation": "<please mention the relation selected from above. Please select only one of the terms.>",

"Bloom": "<please mention the Bloom's taxonomy term selected from above. Please select only one of the terms.>"

"Question": "A very challenging question based on a synthesis of the scenario, relation type, Bloom's taxonomy term and implicitly

```

    similar in structure to the examples mentioned above with the
    intent to evaluate a large language model for reasoning
    effectively. Most question contents should be in the scenario
    information shared. As a result, please keep the questions
    concise.",
"A": "<Details of option A.>",
"B": "<Details of option B. Since it is an MCQ type evaluation, we
    must make sure that the question isn't already answered in other
    options or in other questions.>",
"C": "<Details of option C. Since it is an MCQ type evaluation, we
    must make sure that the question isn't already answered in other
    options or in other questions.>",
"D": "<Details of option D. Since it is an MCQ type evaluation, we
    must make sure that the question isn't already answered in other
    options or in other questions.>",
"Answer rationale": "<string describing answer rationale>",
"Correct option": "<A or B or C or D that is the correct option. This
    will be used as the reference, so please make sure it is
    absolutely correct in accordance with the rationale written above
    .>"
<close curly bracket (specific question)>
...
...
<close curly bracket (all questions)>
<close curly bracket>

Please only generate the json dictionary exactly as specified. Please
    don't generate anything else.
Nothing else. Nothing else. Nothing else. Please do not add any tags

```

of any sort , just the formatting shown here .

Listing 4.7: *QA structure prompt for the spatial reasoning QA data generation task. The examples are generated after several iterations with another LLM and are based on the respective works cited in Section 4.3.1*

Here is some additional information about multiple-choice question formation as well as Bloom’s taxonomy:

A multiple-choice question is a collection of three components aimed at testing a student’s understanding of a certain topic , given a particular context of what the student is expected to know. The topic , as well as the context of the topic , will be provided in order to generate effective multiple-choice questions. The three components of a multiple-choice question are as follows: a Stem, a Correct Answer, and two Distractors. There must always be only one correct answer and only two distractors.

The stem refers to the question the student will attempt to answer , as well as the relevant context necessary in order to answer the question. It may be in the form of a question , an incomplete statement , or a scenario. The stem should focus on assessing the specific knowledge or concept the question aims to evaluate.

The Correct Answer refers to the correct , undisputable answer to the question in the stem.

A Distractor is an incorrect answer to the question in the stem and adheres to the following properties. Use “None of the Above” or “All of the Above” style answer choices sparingly. These answer choices have been shown to , in general , be less effective at measuring or assessing student understanding.

Multiple-choice questions should be clear, concise, and grammatically correct statements. Make sure the questions are worded in a way that is easy to understand and does not introduce unnecessary complexity or ambiguity. Students should be able to understand the questions without confusion. The question should not be too long, and allow most students to finish in less than five minutes. This means adhering to the following properties.

Blooms' Taxonomy and Action Verbs:

Multiple-choice questions must be well aligned to the learning objectives they are intended to assess students' knowledge on. This implies that they must assess skills at the right cognitive level corresponding to the Bloom's taxonomy categorization of the learning objective. Bloom's Taxonomy offers a framework for categorizing the depth of learning, and it provides guidance on selecting appropriate action verbs when writing learning objectives. Here are the six levels of Bloom's taxonomy and their definitions:

Remember – This level involves retrieving, recognizing, and recalling relevant knowledge from long-term memory.

Understand – At this level, learners construct meaning from oral, written, and graphic messages through interpreting, exemplifying, classifying, summarizing, inferring, comparing, and explaining. Apply – This level requires learners to carry out or use a procedure through executing or implementing it.

Analyze – At this level, learners break material into constituent parts, determine how the parts relate to one another and to an overall structure or purpose through differentiating, organizing. Evaluate – This level involves making judgments based on criteria and standards through checking and critiquing.

Create – At this level, learners put elements together to form a coherent or functional whole, or they reorganize elements into a new pattern or structure through generating.

Given information about the traffic related ontology, as well as question formation guidelines and Bloom's taxonomy, would you be able to create multiple-choice questions for EACH TYPE OF INFERENCE provided below that fulfill the MCQ guidelines as well as Bloom's taxonomy? The below example shows the type of inference relation as well as an example of the types of questions asked with that particular relation. Create one question for each type of relation. For option D, make sure the option is similar to the answer to the question, however incorrect. Also, create an answer sheet with the answers to the questions.

Here are some examples. I want you to consider them, adapt the structure within these examples.

TIME RELATIONS:

Domain Knowledge:

A: For urban driving tests, regulations require the AV to maintain a following time of at least ____ under typical city traffic.

B: I'll set the adaptive cruise control accordingly.

Options:

(correct) 2 seconds

(incorrect) 1 second

(correct) 1.5 seconds

(incorrect) 0.5 seconds

Comparison:

A: In dry conditions , the emergency-braking test stopped the vehicle in 28 meters . Under wet conditions , it stopped in 35 meters .

B: Which scenario yields the shorter braking distance?

A: ____.

B: Exactly .

Options :

(correct) dry conditions

(incorrect) snow

(correct) scenario A

(incorrect) scenario B

Arithmetic :

A: Our suburban test route is 20 km long . We plan 4 laps to simulate extended operation .

B: The total distance covered will be ____.

A: Correct .

Options :

(incorrect) 80 km/h

(correct) 80 km

(correct) 80 000 meters

(incorrect) 100 km

General Commonsense :

A: How long does the lidar sensor need to stabilize after power-up before logging reliable point-cloud data?

B: It typically takes ____.

A: Thanks .

Options :

(correct) 90 seconds

(incorrect) 30 seconds
(correct) 2 minutes
(incorrect) 10 minutes

Others (Configuration):

A: We need to test vehicle-to-infrastructure communication on our prototype. Which protocol should we enable on the roadside unit?

B: Let's switch on the ____.

A: Sounds good.

Options:

(correct) DSRC
(incorrect) Bluetooth
(correct) C-V2X
(incorrect) Wi-Fi

Now we will generate the questions. Please remember that any of the options A or B or C or D can be the correct one and ordering can be an important aspect.

Please shuffle the ordering of the option that is correct. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.

Please format the question and answer data in the form of a dictionary as follows:

<open curly bracket>

"Scenario": "<A scenario description in narrative form based on a subset of the ontology concepts mentioned above. Several questions will be written on the basis of this scenario, so please present the the necessary details accurately. Please describe the scenario as a sequence of events that are unfolding. You do not need to fully cover the scenario concepts that are

mentioned above, only a subset of it. Please ensure that the scenario is plausible.>",

"Questions": <open curly bracket>

"<question number label written as Q1, Q2 and so on>":<open curly bracket>

"Relation": "<please mention the relation selected from above. Please select only one of the terms.>",

"Bloom": "<please mention the Bloom's taxonomy term selected from above. Please select only one of the terms.>"

"Question": "A very challenging question based on a synthesis of the scenario, relation type, Bloom's taxonomy term and implicitly similar in structure to the examples mentioned above with the intent to evaluate a large language model for reasoning effectively. Most question contents should be in the scenario information shared. As a result, please keep the questions concise.",

"A": "<Details of option A.>",

"B": "<Details of option B. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.>",

"C": "<Details of option C. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.>",

"D": "<Details of option D. Since it is an MCQ type evaluation, we must make sure that the question isn't already answered in other options or in other questions.>",

"Answer rationale": "<string describing answer rationale>",

"Correct option": "<A or B or C or D that is the correct option. This will be used as the reference, so please make sure it is absolutely correct in accordance with the rationale written above .>"

<close curly bracket (specific question)>

```

... ,
...
<close curly bracket (all questions)>
<close curly bracket>

Please only generate the json dictionary exactly as specified. Please
    don't generate anything else.
Nothing else. Nothing else. Nothing else. Please do not add any tags
    of any sort , just the formatting shown here.

```

Listing 4.8: *QA structure prompt for the temporal reasoning QA data generation task. The examples are generated after several iterations with another LLM and are based on the respective works cited in Section 4.3.1*

4.4 Experimental Setup

The QA generation involves prompting the LLM GPT-4 following the design explained in section 4.3. The focus of these experiments were to address the following research questions:

- **RQ 1:** How do we generate diverse and effective scenario reasoning question and answer data in a controllable manner using pre-trained LLMs?

We carry out test runs to evaluate the performance of the QA generation process detailed in section III. Each test run consists of a configuration of the following parameters:

- **Prompt Wording:** This includes the wording of specific aspects of the prompt detailed in section III.
- **Ordering:** This pertains to the ordering of the different aspects of the prompt.

- **Ontology Updates:** This includes the number of steps to refine the ontology such that the scenario that the questions are based on have depth.
- **Difficulty Updates:** We feed the LLM outputs back with another prompt to increase the difficulty of the generated questions.
- **Reasoning type:** These are covered in Section 4.3 of this chapter.
- **Number of examples provided:** Following the Chain-of-thought prompting process in [146], this parameter details the number of examples.
- **Elements removed from the base prompt:** Following the process of ablation testing, we measure the impact of parts of the prompt being removed.
- **Number of questions generated.**
- **Reasoning hops needed:** Following [111], we categorize reasoning-based questions in terms of reasoning hops needed to solve them. This provides us with a measure of the difficulty of a question.
- **Number of iterations in a single chat context:** Since the LLM used is a *chat completion* model, the sequence of inputs and outputs have a message-based chat format. This, in addition to other advantages of these LLMs, means that we can refine outputs by iteratively generating the same response with some feedback [4]. We test whether this process improves the questions or not.
- **Temperature:** LLMs contain a *temperature* parameter to introduce randomness and creativity in outputs. We evaluate the impact of this parameter, especially on the generation consistency.
- **Pre-trained LLM used to generate questions.**

Scenario:

You are driving an autonomous vehicle on a wet, rainy night. The road is a two-lane highway with a speed limit of 60 mph. Suddenly, a pedestrian walks onto the road from the pedestrian crossing without looking. Your vehicle's safety features kick in and the vehicle automatically applies brakes to avoid a collision.

Questions:

1. (Understand, Causes, 1 Hop)

What can be the immediate result of a pedestrian suddenly walking onto the road from the pedestrian crossing in front of an autonomous vehicle?

- A. The vehicle will increase its speed.
- B. The vehicle will automatically apply brakes.
- C. The vehicle will change its route.
- D. The vehicle will continue at the same speed.

Rationale: Understanding the cause and effect relationship and autonomous vehicle's safety features, it's clear that the vehicle will apply brakes to avoid a collision.

Answer: B

2. (Apply, Causes, 2 Hops)

If the road surface is slick due to rain, what can be the result of an autonomous vehicle suddenly applying brakes?

- A. The vehicle can skid.
- B. The vehicle will increase its speed.
- C. The vehicle will stop immediately.
- D. The vehicle's lights will turn off.

Rationale: Applying the principles of physics and understanding that sudden braking on a wet road can lead to skidding, we can infer the most likely outcome.

Answer: A

Figure 4.4: QA Generation

- **Consistency among multiple generations:** Since the LLMs have randomness in their generation process, getting consistent outputs is difficult. We evaluate this.
- **Solution and explanation generation:** We evaluate the generated questions with and without solution and explanation generation (and the impact of their ordering).

We carry out an adequate number of permutations of this configuration. We observe and record the questions generated by the LLM. An example of one such QA set generation is shown in Figure 4.4.

4.4.1 Controllability and Diversity

One of the advantages of CruzwayQA is a concepts-first and modular QA generation approach. This allows for significant controllability in the generated QA data. We run experiments to evaluate for this by modifying the initial ontology prompt. This results in downstream changes which changes the QA data.

Along these lines, the diversity of the generated questions is along the following pre-defined breadth categories: (1) Reasoning Types. This is a nested category. There are several reasoning relations as mentioned previously within each reasoning category. (2) Ontology considered. (3) Bloom’s taxonomy term considered. In addition, we also evaluate QA data generation along the following depth categories: (1) Number of updates to ontology generated. (2) Number of updates to the QA data such that the generated questions are more difficult.

4.4.2 LLM performance

We carry out test runs to evaluate the performance of pre-trained LLMs in answering the questions generated by the LLM GPT 4.1. We manually grade the explanations for good reasoning skills. In addition, we record the accuracy of the options selected by pre-trained LLMs while attempting a certain number of questions at a time along the question diversity breadth and depth pre-defined categories as mentioned above, with a focus on LLMs *other than* the one that generated these questions. A summary of this evaluation is written in the next section. To increase the efficiency of the test runs, we make use of an *LLM playground* feature of the *LiteLLM* package that allows us to evaluate the response of multiple pre-trained LLMs simultaneously [63]. Examples of these are shown in Figure 4.5.

4.5 Results

We used the pre-trained LLM GPT-4 for evaluating the QA generation process. By carefully changing the experiment parameters detailed in the previous section, we gained valuable insights regarding effective prompting strategies for generating

User Input

Enter your prompt here:

Read the autonomous vehicle test scenario below and answer the questions.

Scenario:

You are driving an autonomous vehicle on a wet, rainy night. The road is a two-lane highway with a speed limit of 60 mph. Suddenly, a pedestrian walks onto the road from the pedestrian crossing

Submit

Model Outputs

gpt-3.5-turbo	gpt-4	command-r	command-nightly	gemini/gemini-pro
1. B. The vehicle will automatically apply brakes. 2. A. The vehicle can skid. 3. A. The vehicle can stop in the	1. B. The vehicle will automatically apply brakes. 2. A. The vehicle can skid. 3. A. The vehicle can stop in the middle of the road	Here are the answers to the questions: 1. B: The vehicle will automatically apply brakes. 2.	Here are the answers to the questions based on the given autonomous vehicle test scenario: 1. B. The vehicle will automatically	1. B. The vehicle will automatically apply brakes. 2. A. The vehicle can skid. 3.

Figure 4.5: Multi-LLM Evaluation

high-quality QA sets, correctness and consistency in generation, and managing redundant questions and answer choices. These insights are detailed below.

The concept of alignment of LLM outputs with respect to user inputs is covered by Kenton et. al [103] [69]. We analyse the LLM outputs for qualitative alignment below.

4.5.1 Alignment

The elements within the generated questions or QA pairs were *nearly always consistent with the instructions provided* in the prompts. The scenarios generated were plausible and made use of the generated ontology. It must be noted though that multiple, simultaneous instructions about maintaining counts were difficult for the LLM to follow. For example, in one test run, we instructed the LLM to generate 10 QA pairs with a certain number from each Bloom’s taxonomy level and a certain number of questions for 1, 2, and 3 reasoning hops. GPT-4 couldn’t maintain all three simultaneous counts correctly in any of the test runs at any of the tested temperature levels. At the same time, the LLM was able to follow multiple, simultaneous instructions regarding *the structure of text generation*. In our prompt, we require GPT-4 to generate multiple choice questions with four options, follow Bloom’s taxonomy, and follow the reasoning examples which it carried out quite well.

In line with the findings by Wei et al. [146], examples of reasoning were crucial for improving the generation process. We observed that removing examples of reasoning and simply instructing it to follow a reasoning pattern either reduced the questions requiring reasoning or removed them completely. The more the number and relevance of given examples, the better the generation. This process has been defined as *in-context learning*, which is a property that emerges as we scale

from small to large language models [145]. In our experiments, at low-temperature values, we observed GPT-4 learning much subtler patterns within examples such as the answer choice which is supposed to be the correct answer and the writing style of answer choices such as the incorrect answer choices for commonsense reasoning were generally implausible. Since the temperature parameter introduces a certain degree of randomness in the next word selection, increasing this value generally reduced *overfitting* to the given examples. Along a similar line, we observed that adding *incorrect* examples e.g. questions with random words or characters for answer choices resulted in a significant reduction in QA writing quality and a significant increase in redundant questions and answer choices.

The presence of Bloom’s taxonomy terms (adapted from Doughty et. al [36]) significantly increased the *depth* of the QA pairs. When removed, the questions did follow the reasoning examples but only superficially, typically concerning the attributes of the scenario elements. Here, a significant insight would be that Bloom’s taxonomy terms in addition to good examples with implicit patterns are crucial for high-quality QA generation. Regarding a few categories though, questions could often belong to multiple categories since a plausible rationale could be generated for each one.

Mentioning the number of reasoning hops required wasn’t adequate in generating complex and challenging questions. However, when examples were provided, the LLM was able to generate complex questions. Increasing complexity of the questions though would often result in multiple correct options being plausible solutions or incorrect options being generated. A significant insight here would be that more examples of challenging questions with complex reasoning hops are needed in addition to a review process.

One significant need in this effort was to have reproducible results. This is

challenging since LLMs are probabilistic models with significant randomness (with the presence of the *temperature parameter*). We observed redundancy in most cases in either the questions generated or the options provided for the questions. In some cases, since questions were on similar topics, some would provide facts that would answer a previous question. Increasing the temperature parameter and *specifying per question structure* would significantly reduce redundancy and make the QA set more diverse. However, with increased temperature, the same prompt in a new chat context would generate completely different scenarios and questions thereby not allowing for experiments to be reproduced at a concrete level. At an abstract level, this is still possible. At a zero temperature value, the generated scenarios and questions would have an increased consistency across new chat contexts but the relation between prompt and generated data still wouldn't be deterministic. Other features such as the OpenAI API's *reproducible outputs* provide the option to ensure increased determinism [6]. However, we weren't able to make use of this *seed* parameter within the *guidance* package that we were using.

4.5.2 Controllability and Diversity

The generation of the QA data is along pre-defined breadth and depth categories as mentioned in the previous section. We present selected results of the generated QA data below. In Listing 4.9, we present questions generated for each of the Bloom's taxonomy terms and deductive reasoning relations. For brevity, in later results reported below, we only present questions for one Bloom's taxonomy term and reasoning type as stated in previous sections.

```
"Scenario ": "An autonomous vehicle is approaching a busy intersection
. The car detects a cyclist performing lane splitting to move
```

between slow-moving vehicles, while several pedestrians are waiting at the curb for the crosswalk signal. Meanwhile, a vehicle ahead signals right before executing a lane change into the rightmost lane. The autonomous vehicle must determine how to safely proceed, ensuring it does not interfere with the cyclist, yields appropriately to pedestrians when the light changes, and adapts to the positions and actions of nearby vehicles.",

```

"Questions": {
  "Q1": {
    "Relation": "Substitution",
    "Bloom": "Evaluate",
    "Question": "Assume the vehicle ahead occasionally
disables its lights entirely, including braking
indicators, while executing abrupt and late lane
changes in dense fog. How should the autonomous
vehicle's perception and prediction systems re-
prioritize sensor weighting and intent inference, and
which advanced counterfactual reasoning models could
help mitigate risks posed by intentional or sensor-
impaired signaling in these conditions?",
    "A": "It should emphasize redundant sensing (radar, lidar
, camera) while incorporating counterfactual intent
modeling'simulating multiple plausible maneuvers for
ill-sigaled vehicles'to predict risk under adverse
or deceptive conditions.",
    "B": "It should ignore intent cues and only react to
physical collisions.",
    "C": "It should rely solely on standard lane-keeping
algorithms.",
    "D": "It should treat the vehicle as an obstacle and
reroute regardless of intersection rules.",

```

```

    "Answer rationale": "Redundant, scenario-driven reasoning
        and simulation increase robustness to intentional
        deception or impaired sensors, especially in low-
        visibility situations.",
    "Correct option": "A"
},
"Q2": {
    "Relation": "Inference from Rule",
    "Bloom": "Analyze",
    "Question": "Given local law stipulates yielding to any
        pedestrian who physically interrupts the curb line '
        even briefly or unintentionally 'and the AV detects a
        group containing both adults and unsupervised
        children displaying unpredictable movement while
        simultaneously being boxed in by vehicles with
        varying stopping distances, how must the AV's policy
        reconcile strict legal compliance with dynamic risk
        assessment, and what measures could be prioritized to
        minimize liability and harm?",
    "A": "Initiate a layered risk response: prioritize
        maximum deceleration within physical safety limits,
        communicate intent with external indicators, and
        prepare for evasive maneuvers, applying conservative
        thresholds for pedestrian risk including anticipated
        child unpredictability.",
    "B": "Default to the average behavior of adult
        pedestrians and ignore children unless they enter the
        lane.",
    "C": "Proceed if at least two adjacent vehicles do so,
        assuming collective safety.",
    "D": "Yield only when no vehicles are present behind the
        AV.",

```

```

    "Answer rationale": "A policy blending strict legal
        compliance and proactive hazard anticipation ,
        especially regarding child pedestrians , most reduces
        harm and legal exposure." ,
    "Correct option ": "A"
},
"Q3": {
    "Relation": "Further Specification or Conjunction",
    "Bloom": "Evaluate",
    "Question": "If updated high-frequency data shows the
        cyclist swerves sharply left to bypass an obstruction
        , immediately triggers an abrupt deceleration , and
        then signals intent to cut across right 'while a
        secondary cyclist approaches from the opposite curb
        at speed 'how should the AV's multi-agent trajectory
        planner reconcile conflicting trajectories , and
        determine priority among vulnerable users for a safe
        right merge?",
    "A": "Deploy multi-agent predictive models to
        probabilistically evaluate possible future
        trajectories for all cyclists , allocating highest
        priority to those with least maneuvering space or
        greatest relative velocity risk , and adjust the merge
        plan to allow dynamic yielding or re-routing as safe
        gaps emerge." ,
    "B": "Merge after the closest cyclist passes , ignoring
        farther threats." ,
    "C": "Signal and wait for both cyclists to pass ,
        regardless of timing or traffic impact." ,
    "D": "Abort the merge entirely if any uncertainty exists
        ." ,

```

```

    "Answer rationale": "Hierarchical, probabilistic planning
        allows continuous risk balancing among multiple,
        dynamically behaving cyclists, preserving safety and
        operational flow.",
    "Correct option": "A"
},
"Q4": {
    "Relation": "Infer Class from Properties",
    "Bloom": "Apply",
    "Question": "Suppose the AV's sensors detect a user
        alternating between pedal-powered and throttle-
        assisted motion, executing abrupt starts and rapid
        stops, sometimes carrying a large item that alters
        silhouette and thermal signature. How should the
        classification algorithm exploit temporal and
        multimodal data for type disambiguation, and what
        precautionary adaptations in AV behavior does this
        necessitate compared to standard cyclist or scooter
        detection?",
    "A": "Leverage time-series analysis across kinematics,
        thermal, and object shape sensors to assign dynamic,
        context-sensitive class probabilities; adapt
        following and passing distances using the riskiest
        plausible object type, and plan for unpredictable
        maneuvers or load interference.",
    "B": "Classify the user based on the most recent observed
        state.",
    "C": "Treat all such users as cyclists, ignoring assisted
        -mobility risks.",
    "D": "Override all dynamic detection and enforce a
        complete stop if the object class is ambiguous.",

```

```

    "Answer rationale": "Integrating diverse signals over
        time enables context-aware object classification and
        risk-adaptive response, crucial for ambiguous micro-
        mobility users.",
    "Correct option": "A"
},
"Q5": {
    "Relation": "Property Inheritance",
    "Bloom": "Analyze",
    "Question": "If among several pedestrians one
        intermittently leaves and rejoins the group while
        making erratic backward steps, and another maintains
        direct line-of-sight with the vehicle while holding a
        child's hand, how should the AV adapt incursion
        prediction logic using group dynamics and social
        context cues, and what minimum safety thresholds
        should override standard yield behavior?",
    "A": "Model group cohesion and social dynamics to
        heighten incursion risk for both individuals'
        anticipating sudden group-driven crossing; enforce
        extended stopping distances and cautious speeds until
        all group members commit to waiting or crossing
        clearly.",
    "B": "Yield only to the individual with the child,
        regardless of group behavior.",
    "C": "Assume the erratic pedestrian is unlikely to cross
        while the group is waiting.",
    "D": "Resume normal speed if most pedestrians remain
        stationary.",
    "Answer rationale": "Accounting for dynamic, group-driven
        behaviors and their social influence on individual
        movement reduces risk and ensures equitable safety.",

```

```

    "Correct option": "A"
  },
  "Q6": {
    "Relation": "Sequential Inference",
    "Bloom": "Analyze",
    "Question": "Imagine the AV receives simultaneous alerts:
      an emergency vehicle weaving through traffic in a
      nearby lane triggers sudden braking cascades ahead,
      while a bus unexpectedly pulls out from curbside,
      partially blocking the right merge path, as a
      delivery drone descends into the intersection
      airspace. What sequential, real-time contingency
      hierarchy should the AV execute to maintain safety
      for all road and airspace users, and what systems
      redundancies are critical under such complex, multi-
      vector threats?",
    "A": "Immediately prioritize yielding to emergency
      vehicles and stopping safely out of its path; assess
      and anticipate predicted trajectories of the bus,
      suspending merge actions while monitoring for further
      bus incursion; activate upward-scanning sensors and
      prepare evasive action for the drone descent, relying
      on layered redundancies across perception, V2X, and
      fail-safe braking systems.",
    "B": "Focus solely on the emergency vehicle, ignoring the
      drone and bus.",
    "C": "Proceed with right merge due to bus's slow speed,
      only stopping if a collision is imminent.",
    "D": "Stop in place and power down all active systems
      until the scene clears.",
    "Answer rationale": "A prioritized, sequential, and
      redundant threat response integrating roadway and

```

```

        aerial awareness is essential for complex, high-
        consequence intersections.",
        "Correct option": "A"
    }
}
}

```

Listing 4.9: QA data generated based on Bloom’s taxonomy terms and deductive reasoning relation terms

```

"Scenario": "During a rainy afternoon in an urban area, an autonomous
    vehicle is undergoing a sequence of test maneuvers along a
    multilane boulevard. The test begins with the vehicle cruising at
    a moderate speed before approaching a bus slowing to stop at a
    curbside. As the vehicle nears, a pedestrian group at a crosswalk
    waits for the signal; meanwhile, another pedestrian unexpectedly
    jaywalks several meters ahead, forcing the vehicle to decelerate
    abruptly. Shortly after, a cyclist to the right signals intent
    to merge into the lane where the autonomous vehicle is traveling.
    The vehicle then performs an evasive lane change to avoid a
    large fallen branch, making use of its emergency systems. Sensors
    record close proximity to both the cyclist and a parked vehicle
    during the maneuver.",
"Questions": {
    "Q1": {
        "Relation": "HasPrerequisite",
        "Bloom": "Evaluate",
        "Question": "Given the reduced sensor range and potential
            for reflection artifacts due to heavy rainfall, the
            simultaneous presence of occlusions from large
            vehicles, and a rapidly evolving hazard environment
            with multiple vulnerable road users, what integrated
            decision-making framework should the autonomous

```

vehicle employ to probabilistically balance safety , assertiveness , and ethical risk minimization when evaluating an emergency evasive lane change , especially in scenarios where sensor fusion modules yield ambiguous or conflicting occupancy maps for adjacent lanes?",

"A": "The vehicle should implement a Bayesian or probabilistic risk assessment framework that quantifies and updates the uncertainty of each sensor source in real time. It must conservatively bias decisions toward minimizing harm to vulnerable road users , integrating ethical cost functions , and 'when conflicting sensor data is present 'prioritize actions with mathematically lowest expected risk (such as pausing or running additional detection cycles before committing to a maneuver).",

"B": "The vehicle should assign equal confidence to all sensor reports and use a majority-vote mechanism to decide lane occupancy , immediately proceeding if the majority of sensors indicate the lane is clear to maintain assertive operation.",

"C": "The vehicle should prioritize local responsiveness by triggering the maneuver based on whichever high-confidence sensor (as determined by a fixed threshold) first detects clear space , foregoing further consideration of uncertainty or ethical weighting due to emergency time constraints.",

"D": "The vehicle should delegate the final lane-change decision to an external cloud-based controller with access to historical traffic and weather data , overriding local sensor assessment to optimize maneuver timing and system consistency.",

```

    "Answer rationale": "Effective and safe AV decision-making in complex, uncertain environments demands dynamic probabilistic reasoning under uncertainty, integration of ethical considerations, and explicit risk minimization'especially when real-time sensor fusion yields no clear consensus. Option A best describes such a state-of-the-art safety approach. Relying on simple majority vote (B), fixed-threshold individual sensors (C), or remote override (D) neglects both context-specific uncertainty and real-time risk analysis, which are critical for ethical and robust autonomous maneuvering in hazardous, occluded situations.",
    "Correct option": "A"
  }
}

```

Listing 4.10: QA data generated based on the ‘evaluate’ Bloom’s taxonomy term and one commonsense relation term

```

"Relation": "Nephew-Style (Indirect Sub-Scenario) Inference",
  "Bloom": "Apply",
  "Question details": {
    "Scenario": "During a comprehensive urban autonomy evaluation, Vehicle A2 initiated a ProtectedLeftTurn maneuver at a signalized intersection, closely following behind Vehicle M9, which had just completed a MultiLaneChange before entering the turn bay. In the same environment, Pedestrian P5 began a CrosswalkCrossing assisted by a mobility device, while Bicycle B3 performed a WrongWayRiding maneuver against the traffic flow. Meanwhile, OnRampMerge operations were being conducted by Vehicle T4, coordinating with other vehicles for a

```

```

MeteredRampMerge onto the main expressway. Parallel to
these events, Driver D7 in another car exhibited signs of
DistractedDriving as observed by traffic monitors.",
"Question": "If a DistributedAutonomyValidation framework
coordinates multiple completely independent subsystems,
and an integration fault in TrafficFlow's OnRampMerge is
escalated to a global monitor that only tangentially logs
(not halts) execution in ParkingSystem's ParallelPark,
how is the relationship between ParallelPark and
OnRampMerge's error lineage best characterized in this
architecture?",
"A": "ParallelPark is an orthogonal module with passive
indirect awareness, but immune to functional error
propagation from OnRampMerge.",
"B": "ParallelPark will actively receive and handle
OnRampMerge errors in real time.",
"C": "ParallelPark is a descendant process within OnRampMerge
's chain.",
"D": "ParallelPark directly supervises OnRampMerge and
mediates its errors."

```

Listing 4.11: QA data generated based on ‘apply’ Bloom’s taxonomy term and one inductive reasoning relation term

```

"Relation": "Relative positions",
"Bloom": "Evaluate",
"Question details": {
  "Scenario": "An autonomous vehicle is driving on a four-lane
city road approaching an intersection during daylight
hours. The vehicle is in the second lane from the left.
Ahead, a slower delivery truck is driving in the same
lane. To the right of the vehicle, a cyclist is riding
close to the curb. In the far-left lane, a sports car is

```

```

preparing to make a protected left turn at the upcoming
intersection. On the sidewalk to the right, a pedestrian
is waiting at the crosswalk while looking at their mobile
device. The vehicle's onboard sensors detect a traffic
light turning yellow as it approaches the intersection.
There is a parallel parked car just beyond the
intersection on the right side of the street.",
"Question": "If all human-controlled vehicles and vulnerable
road users are anticipated to react sub-optimally to the
traffic signal change (e.g., late stopping, accelerating
unexpectedly), which target lane placement should the
autonomous vehicle compute as lowest overall exposure to
compound dynamic risk factoring likely late crosswalk
entry by the pedestrian and erratic path deviations from
the cyclist and sports car based on a probabilistic risk
heatmap for the entire intersection approach?",
"A": "Far-left lane",
"B": "Third lane from the left (one right of current position
)",
"C": "Second lane from left (current position)",
"D": "Far-right lane"

```

Listing 4.12: QA data generated based on ‘evaluate’ Bloom’s taxonomy term and one spatial reasoning relation term

```

"Relation": "Anticipatory Multi-Agent Response Planning",
"Bloom": "Evaluate",
"Question details": {
  "Scenario": "During an urban autonomous vehicle test, the AV
approaches a signalized intersection. A pedestrian is
observed waiting at the crosswalk. As the light turns green,
the AV prepares to execute a left turn (ProtectedLeftTurn)
while monitoring nearby traffic. A cyclist is detected in the

```

leftmost lane, moving in the same direction as the AV. Before making the turn, the AV activates its turn signal and reduces speed to a complete stop, waiting for the pedestrian to cross. After the pedestrian crosses safely, the AV initiates a ProtectedLeftTurn and merges onto a two-lane road, accelerating to traffic speed. Shortly thereafter, it encounters a slow-moving vehicle ahead and assesses the possibility of a SafeOvertake maneuver. The AV decides to overtake the vehicle while maintaining safe distance from a cyclist now riding closer to the right curb. The test monitors AV compliance with required safety measures, signaling, and response to vulnerable road users.",

"Question": "Assume after the overtake, a sensor anomaly causes the AV's main LiDAR to briefly malfunction, just as a cyclist at the curb suddenly drops an object and veers unpredictably toward traffic. Simultaneously, a heavy delivery truck is following the AV at close range and vehicle-to-vehicle communication is intermittent. How should the AV employ redundancy, predictive modeling, and risk prioritization across multiple actors to minimize potential harm, and what documentation steps are necessary for regulatory compliance post-incident?",

"A": "Immediately prioritize sensor redundancy by switching to backup modalities (radar/camera), estimate trajectories for both cyclist and following truck using conservative prediction envelopes, initiate an emergency deceleration accompanied by maximum hazard signaling, continuously broadcast intent over all available V2V channels, and comprehensively log sensor anomalies and all AV decisions for later regulatory review.",

"B": "Rely solely on the most recent sensor data and wait for primary LiDAR restoration before making any maneuver.",

```

"C": "Focus exclusively on the cyclist's path, disregarding
      trailing vehicle dynamics due to sensor uncertainty.",
"D": "Accelerate sharply to vacate the affected zone, even if it
      puts the cyclist and following truck at greater risk."
},

```

Listing 4.13: QA data generated based on ‘evaluate’ Bloom’s taxonomy term and one temporal reasoning relation term

```

"Ontology": "...
dot digraph Ontology {
TrafficActivity -> DrivingActivity;
TrafficActivity -> HumanActivity;
TrafficActivity -> MovingActivity;
DrivingActivity -> ChangeLane;
DrivingActivity -> Overtake;
DrivingActivity -> MakeATurn; DrivingActivity -> Park;
DrivingActivity -> FollowVehicle;
DrivingActivity -> EnterIntersection; DrivingActivity -> EmergencyStop
; MakeATurn -> MakeALeftTurn;
MakeATurn -> MakeARightTurn; MakeATurn -> MakeAUturn;
MakeALeftTurn -> SignalLeftTurn;
MakeALeftTurn -> YieldToPedestrian;
HumanActivity -> PedestrianActivity;
HumanActivity -> DriverActivity;
HumanActivity -> RiderActivity;
PedestrianActivity -> CrossStreet;
PedestrianActivity -> WaitAtCrosswalk;
PedestrianActivity -> Jaywalk; PedestrianActivity ->
UseMobileWhileWalking; DriverActivity -> Yield;
DriverActivity -> Honk; DriverActivity -> UseTurnSignal;
DriverActivity -> CheckBlindSpot;
MovingActivity -> Accelerate;

```

```

MovingActivity -> Decelerate;
MovingActivity -> Stop; MovingActivity -> Merge;
MovingActivity -> AvoidObstacle;
AvoidObstacle -> SlowDownForObstacle;
AvoidObstacle -> Swerve;",
  "Relation": "General Commonsense",
  "Bloom": "Create",
  "Question details": {
    "Scenario": "An autonomous vehicle (AV) is navigating a busy
      urban street during mid-morning hours. The AV is
      following another car while approaching an intersection
      controlled by traffic lights. As it nears the
      intersection, a pedestrian steps onto the crosswalk,
      waiting to cross. At the same time, another vehicle in
      the adjacent lane signals and begins to change lanes in
      front of the AV. The AV detects the pedestrian, slows
      down, and stops to yield. Once the pedestrian crosses,
      the AV checks its blind spot before proceeding through
      the intersection, making a right turn. During this
      maneuver, it briefly accelerates to merge with the
      ongoing traffic on the next street."
  },

```

Listing 4.14: Ontology generated after three update steps and the corresponding scenario generated. The QA data is based on the generated scenario. All generation is by GPT 4.1. The ontology generation is based on the prompts presented in Listings 4.1 and `reflst:ontology_refining_prompt`

```

"Ontology": "dotdigraph Ontology {
DrivingActivity -> ChangeLane;
DrivingActivity -> Overtake;
DrivingActivity -> MakeATurn;

```

```

DrivingActivity → Park;
DrivingActivity → EmergencyManeuver;
DrivingActivity → Yield;
DrivingActivity → MergeOntoHighway;
ChangeLane → SignalBeforeLaneChange;
ChangeLane → BlindSpotCheck;
ChangeLane → DoubleLaneChange;
DoubleLaneChange → HighSpeedDoubleLaneChange;
MakeATurn → MakeALeftTurn;
MakeATurn → MakeARightTurn;
MakeATurn → UTurn;
MakeALeftTurn → ProtectedLeftTurn;
ProtectedLeftTurn → LeftTurnAtIntersection;
LeftTurnAtIntersection → DetectOncomingTraffic;
    DetectOncomingTraffic → PredictGapAcceptance;
MakeARightTurn → RightTurnAtIntersection;
RightTurnAtIntersection → DetectPedestrianDuringTurn;
    EmergencyManeuver → HardBrake;
EmergencyManeuver → EvasiveSteer;
EmergencyManeuver → AvoidObstacle;
AvoidObstacle → AvoidPedestrian;
AvoidObstacle → AvoidVehicle;
AvoidVehicle → PredictVehicleIntent;
AvoidPedestrian → PredictPedestrianCrossing;
Park → ParallelPark;
Park → PerpendicularPark;
ParallelPark → TightParallelPark;
Yield → YieldToPedestrian;
YieldToPedestrian → DetectYieldSign;
MergeOntoHighway → RampMeteringMerge;
HumanActivity → PedestrianActivity;
HumanActivity → DriverActivity;

```

```

PedestrianActivity -> Cross;
PedestrianActivity -> WalkAlongRoad;
PedestrianActivity -> WaitAtCurb;
PedestrianActivity -> UseMobilityAid;
Cross -> CrossAtCrosswalk;
Cross -> Jaywalk;
CrossAtCrosswalk -> WaitForSignal;
WaitForSignal -> RecognizeTrafficLight;
WalkAlongRoad -> WalkOnShoulder;
WalkAlongRoad -> DistractedWalking;
WalkAlongRoad -> EnteringFromHiddenArea;
UseMobilityAid -> WheelchairUser;",
  "Relation": "General Commonsense",
  "Bloom": "Create",
  "Question details": {
    "Scenario": "An autonomous vehicle is navigating an urban
      environment on a weekday afternoon. The vehicle initiates
      a right turn at a busy intersection after detecting a
      green signal. As it begins the turn, the sensors detect a
      pedestrian approaching the crosswalk, who pauses at the
      curb. The AV yields to the pedestrian. Just after
      completing the turn, the AV identifies a double-parked
      vehicle partially blocking its lane and must perform a
      controlled lane change, using its signal in advance and
      verifying the adjacent lane is clear. As the AV moves
      forward, a cyclist suddenly enters from a hidden driveway
      , requiring the vehicle to execute a hard brake to avoid
      a collision."
  }

```

```
},
```

Listing 4.15: Ontology generated after five update steps and the corresponding scenario generated. The QA data is based on the generated scenario. All generation is by GPT 4.1. The ontology generation is based on the prompts presented in Listings 4.1 and `reflst:ontology_refining_prompt`

```
...
"Question details": {
  "Scenario": "During an afternoon in a busy urban environment ,
    an autonomous vehicle is tested on a multilane road with
    moderate traffic . The AV approaches an intersection
    where a pedestrian is waiting to cross legally with the
    signal . At the same time , a cyclist is waiting at the
    intersection , preparing to mount their bicycle , while a
    human-driven car in the adjacent lane signals a right
    turn but begins to use a mobile device . As the light
    turns green , the pedestrian begins to cross at the
    crosswalk as the vehicle in the adjacent lane initiates a
    right turn , and the cyclist mounts their bicycle and
    enters the lane . The AV must decide how to proceed safely
    during this sequence of events ." ,
  "Question": "While the pedestrian with dual aids is still
    crossing , the cyclist abruptly shifts direction and a
    delivery robot enters the intersection from an unmarked
    curb cut , as the human driver 's distraction persists .
    Design an integrated risk assessment protocol for the AV
    that quantitatively fuses real-time multi-agent
    uncertainty (including occluded actors and non-human road
    users) , dynamically adapts clearance policies , and
    factors in ethical priorities enshrined in both AV
    deployment charters and Vision Zero principles . Which
```

```

        operational mode best fulfills these multilayer criteria
        and why?",
    "A": "Implement a real-time, layered risk fusion engine
        incorporating Bayesian intent prediction for all detected
        /occluded actors (human and non-human), set ultra-
        conservative dynamic clearance margins that adapt to
        compounding uncertainties, and adopt a fail-safe halt
        policy triggered by unpredictable cross-domain risk, with
        transparent external communication of paused status and
        justification.",
    "B": "Rely on static actor-based risk tables and ignore non-
        human entrants unless physically blocking the AV's path
        .",
    "C": "Continue with preplanned timings and fixed clearance
        zones, assuming non-human agents yield or stop if
        encountered.",
    "D": "Prioritize urban traffic throughput; enforce minimal
        stops unless a human actor is in direct collision
        proximity."
},

```

Listing 4.16: Keeping the ontology refine level as 2

```

...
"Question details": {
    "Scenario": "An autonomous vehicle is navigating an urban
        environment during early morning traffic. The AV is following
        a sedan ahead at a safe distance. As it approaches an
        intersection, the sedan activates its left turn signal and
        begins to slow down. Meanwhile, a cyclist riding in the same
        direction dismounts at the corner, and a pedestrian waits at
        the crosswalk, preparing to cross once the signal changes.
        The AV system must decide whether to yield, follow, or

```

attempt a lane change to the right. The weather is clear, and all sensors are operating normally.",

"Question": "Imagine the traffic light turns green as a pedestrian unexpectedly dashes diagonally across the intersection, a second cyclist surges from the sidewalk into the AV's intended turn path, and an approaching fire engine is now confirmed by audio triangulation but remains visually occluded. Describe the suite of advanced contingency protocols 'combining explicit external signaling, standardized V2X negotiations, selective intent broadcasting, and adaptive space-sharing gestures' that the AV should deploy to minimize liability, maintain legal/ethical compliance, and proactively foster social trust among all participants amid overlapping legal and practical risks.",

"A": "Simultaneously engage integrated communication 'including programmable LED intent displays, AV-to-infrastructure broadcasts alerting emergency vehicle proximity, context-responsive external speaker prompts, and visual projections to inform VRUs of yielding/intent' while adaptively yielding, reducing speed to a crawl, mapping opt-out safety zones for all ambiguous agents, recalibrating planning based on both codified law and dynamic social cues, and logging all interactions for transparent post-incident analysis.",

"B": "Proceed with right-of-way maneuvers immediately upon light change, pausing only if direct contact risk is detected.",

"C": "Rely exclusively on V2X broadcasts without human-readable cues, to minimize distraction for VRUs and reduce onboard system complexity.",

"D": "Temporarily disable intent communication in high-conflict cases to avoid sending mixed signals and focus only on maximizing physical clearance."

```
} ,
```

Listing 4.17: Keeping the ontology refine level as 2

4.5.3 LLM performance

We generate around 10 questions at a time along each category and measure the accuracy of the options selected by the LLMs selected across model families and model sizes. These results are summarized in Table 4.1. The dataset is reasonably effective e.g. we can observe that spatial reasoning is a challenge for all categories of LLMs. At the same time, we can observe that the dataset generated is not very challenging and various measures (such as iterative refinement of questions or manual reviews) can be taken to increase the QA data difficulty. The presence of the scenario information with the question does provide additional context about the event and as we can observe has a positive effect on the answer accuracy.

4.6 Conclusion

In this chapter, we presented CruzwayQA, a concepts-first, modular approach for generating synthetic QA data for the purpose of evaluating the scenario reasoning ability of LLMs. With CruzwayQA, we are able to generate questions which are a synthesis of concepts from the OpenXOntology standard, an MCQ generation approach, concepts from Bloom’s taxonomy and ones which evaluate for a variety of reasoning styles such as deductive, inductive, and most importantly spatial and temporal reasoning. This system is highly controllable since we can customize the ontology and focus on a variety of features within each aspect of the module described. In addition, we systematically evaluate the question and

option generation ability in CruzwayQA and the effectiveness of the questions on LLMs across model families and sizes. Through this, we determine that providing examples and Bloom’s taxonomy terms was crucial to generating questions that align with the instructions. While the QA data generated is reasonably effective in evaluating LLMs, steps can be taken to increase the difficulty of questions. In the future, we aim to increase the realism of the generated questions, possibly by combining with more realistic QA datasets such as the Waymo Open Motion Dataset - Reasoning, which is considered in Chapter 5.

LLM Evaluated	Scenario Reasoning Category (Set)	Presence of Scenario Information	Accuracy for given set (percentage)
GPT 4o-mini	Commonsense Reasoning (10)	Absent	100
GPT 4o-mini	Commonsense Reasoning (8)	Present	100
GPT 4o-mini	Deductive Reasoning (10)	Absent	90
GPT 4o-mini	Deductive Reasoning (9)	Present	100
GPT 4o-mini	Spatial Reasoning (10)	Absent	80
GPT 4o-mini	Spatial Reasoning (10)	Present	80
GPT 4o-mini	Temporal Reasoning (10)	Absent	100
GPT 4o-mini	Temporal Reasoning (10)	Present	100
GPT 4o-mini	Inductive Reasoning (9)	Absent	77.77
GPT 4o-mini	Inductive Reasoning (9)	Present	100
Deepseek V3	Spatial Reasoning (10)	Absent	70
Deepseek V3	Spatial Reasoning (10)	Present	90
Deepseek V3	Temporal Reasoning (10)	Absent	100
Deepseek V3	Temporal Reasoning (5)	Present	100
Deepseek V3	Deductive Reasoning (10)	Absent	80
Deepseek V3	Deductive Reasoning (10)	Present	100
Deepseek V3	Inductive Reasoning (10)	Absent	90
Deepseek V3	Inductive Reasoning (3)	Present	100
Deepseek V3	Commonsense Reasoning (3)	Absent	100
Deepseek V3	Commonsense Reasoning (10)	Present	100
o1	Deductive Reasoning (10)	Absent	100
o1	Deductive Reasoning (10)	Present	100
o1	Inductive Reasoning (10)	Absent	100
o1	Temporal Reasoning (10)	Absent	80
o1	Commonsense Reasoning (10)	Absent	100
o1	Commonsense Reasoning (3)	Present	100
o1	Spatial Reasoning (10)	Absent	90
o1	Spatial Reasoning (10)	Present	100

Table 4.1: Evaluation of LLMs for the sets of 10 questions of generated QA data along the pre-defined breadth and depth categories as stated in Section 4.4.

Chapter 5

Cruzway Scenario Generator

5.1 Introduction

Addressing challenges of LLM-based scenario generation requires addressing challenges regarding scenario reasoning. LLM-based scenario generation holds the promise of improving the controllability and interpretability of data-driven methods while maintaining their realism. Zhao et. al present five challenges for the application of LLMs for scenario-based testing of autonomous vehicles [157]. Here, one challenge is that of hallucinations which the authors define as “generated outputs which are unfaithful to inputs.” Specifically, this includes “non-executable code in scenario generation and factually incorrect judgments for safety-critical assessments.” Other challenges include the lack of works involving the creation of scenario databases for reuse and corresponding interfaces that transform scenarios into machine-readable formats. These challenges can be easily mapped to the four AV test scenario generation challenges by Ding et. al which are addressed throughout this dissertation; hallucinations in scenario generation can be mapped to scenario realism, controllability and diversity while database creation can be mapped to transferability. Fundamentally, the concept of hallucinations in sce-

nario generation can be connected to the concept of scenario reasoning defined in Chapter 4 as a *synthesis of non-exhaustive subset of reasoning structures such as deductive, inductive, spatial, temporal as well as System 1 and 2 reasoning with concepts relevant to AV test scenarios such as the ones in the ASAM OpenXOntology [10]*. Each reasoning structure involves unique relations between successive statements as seen in *EntailmentBank* [29] or between concepts in the statements as seen in *CommonsenseQA* [134].

Scenario Reasoning

A synthesis of a non-exhaustive subset of reasoning structures such as deductive, inductive, spatial, temporal as well as System 1 and 2 reasoning with concepts relevant to AV test scenarios such as the ones in the ASAM OpenXOntology.

More specifically, dynamic scenarios contain traces of *System 2 variants* of scenario reasoning. System 2 reasoning, defined by Kanheman constitutes a more explicit, structured and deliberate form of reasoning compared to a more instinct-driven, implicit form of reasoning defined as System 1 reasoning [?]. Within the context of LLMs, we can observe the progress from System 1 to System 2 reasoning with the advent of reasoning language models such as *OpenAI o1* as presented in a survey on this topic by Li et. al [81]. Dynamic scenario representations in tools such as the scenario behavior tree representation in Carla Scenario Runner and standardized scenario formats such as OpenSCENARIO [23] [11] are structurally and functionally similar to *domain* and *problem* representations in *model-based task planning*. In both forms of representations, the agent actions are defined as *conditions* followed by *effects*, and at a larger scale the a sequence of actions are organized into *behaviors*. An example of a model-based task planning domain

file (expressed in the Planning Domain Definition Language (PDDL) format) is presented in Listing 5.1 and an example of an truncated OpenSCENARIO file is presented in 5.2. We can consider this sequence of actions as an example of System 2 reasoning. Since the ontology is consistent with the OpenXOntology suggested in the scenario reasoning definition above, we can consider that these scenario definitions contain *traces of System 2 variants of scenario reasoning*.

```
(define (domain autonomous_vehicle_intersection)
  (:requirements :strips :typing)
  (:types agent intersection direction)
  (:predicates
    (ego ?a – agent)
    (vehicle ?a – agent)
    (in-intersection ?a – agent ?i – intersection)
    (facing-straight ?a – agent)
    (exiting-intersection ?a – agent ?i – intersection)
    (decelerating ?a – agent)
    (speed-zero ?a – agent)
    (moving-slow ?a – agent)
    (no-stop-sign-on-side ?a – agent ?i – intersection)
    (proceeds-without-stopping ?a – agent ?i – intersection)
    (near-intersection ?a – agent ?i – intersection)
    (heading-opposite ?a – agent ?b – agent)
    (heading-same ?a – agent ?b – agent)
    (in-front-of ?a – agent ?b – agent)
    (on-same-side ?a – agent ?i – intersection)
    (left-of-intersection ?a – agent ?i – intersection)
    (five-way-intersection ?i – intersection)
    (three-stop-signs ?i – intersection)
    (not-moving ?a – agent)
  )
)
```

```

(:action ego-begins-exiting-intersection
  :parameters (?ego - agent ?i - intersection)
  :precondition (and
    (ego ?ego)
    (in-intersection ?ego ?i)
    (facing-straight ?ego)
  )
  :effect (exiting-intersection ?ego ?i)
)

(:action ego-decelerates-to-stop
  :parameters (?ego - agent)
  :precondition (and
    (ego ?ego)
    (moving-slow ?ego)
    (decelerating ?ego)
  )
  :effect (speed-zero ?ego)
)

(:action ego-unimpeded-by-stop-sign
  :parameters (?ego - agent ?i - intersection)
  :precondition (and
    (ego ?ego)
    (facing-straight ?ego)
    (no-stop-sign-on-side ?ego ?i)
  )
  :effect (proceeds-without-stopping ?ego ?i)
)

(:action surrounding-agent0-decelerates-near-intersection
  :parameters (?a0 - agent ?i - intersection)

```

```

:precondition (and
  (vehicle ?a0)
  (near-intersection ?a0 ?i)
  (decelerating ?a0)
)
:effect (moving-slow ?a0)
)

(:action surrounding-agent2-and-3-wait-opposite-ego
:parameters (?a2 - agent ?a3 - agent ?ego - agent)
:precondition (and
  (vehicle ?a2)
  (vehicle ?a3)
  (ego ?ego)
  (heading-opposite ?a2 ?ego)
  (heading-opposite ?a3 ?ego)
)
:effect (and
  (not-moving ?a2)
  (not-moving ?a3)
)
)

(:action surrounding-agent4-waits-ahead-same-direction
:parameters (?a4 - agent ?ego - agent)
:precondition (and
  (vehicle ?a4)
  (ego ?ego)
  (in-front-of ?a4 ?ego)
  (heading-same ?a4 ?ego)
)
:effect (not-moving ?a4)
)

```

```

)

(:action intersection-configuration
  :parameters (?i - intersection)
  :precondition (five-way-intersection ?i)
  :effect (three-stop-signs ?i)
)
)

```

Listing 5.1: A *Planning Domain Definition Language (PDDL)* domain file contains definitions of conditional actions. Each action maps abstract precondition states to effect states. A sequence of these actions constitutes a plan (omitted here). These definitions can be considered as *System 2* variants of scenario reasoning defined above.

```

<Event priority="overwrite" maximumExecutionCount="1" name="Turn
  right">
  <Action name="Lane Change">
    <PrivateAction>
      <LateralAction>
        <LaneChangeAction>
          <LaneChangeActionDynamics dynamicsDimension="distance "
            dynamicsShape="cubic" value="20.0"/>
          <LaneChangeTarget>
            <AbsoluteTargetLane value="-4"/>
          </LaneChangeTarget>
        </LaneChangeAction>
      </LateralAction>
    </PrivateAction>
  </Action>
  <StartTrigger>
    <ConditionGroup>

```

```

<Condition delay="0.0" conditionEdge="rising" name="Relative
  Distance">
  <ByEntityCondition>
    <TriggeringEntities triggeringEntitiesRule="any">
      <EntityRef entityRef="Vehicle 2"/>
    </TriggeringEntities>
    <EntityCondition>
      <RelativeDistanceCondition freespace="false" rule="
        greaterThan" entityRef="Vehicle 1" value="10.0"
        relativeDistanceType="longitudinal"/>
    </EntityCondition>
  </ByEntityCondition>
</Condition>
<Condition delay="0.0" conditionEdge="none" name="Simulation
  Time">
  <ByValueCondition>
    <SimulationTimeCondition rule="greaterThan" value="5.0"/>
  </ByValueCondition>
</Condition>
</ConditionGroup>
</StartTrigger>
</Event>

```

Listing 5.2: *An event within the ASAM OpenSCENARIO standard defining an lane change behavior interaction between two vehicles Vehicle 1 and 2. Here, the dynamic scenario definition is structurally similar to the PDDL vehicle behavior definition in Listing 5.1 as conditional actions*

Combining symbolic systems with LLMs presents an approach to validate and verify LLM generated scenarios by affording syntax and semantic verification along with reasoning validation. This requires modeling the problem for the solver which in the case of dynamic scenarios, is the same as scenario generation. *LLM-*

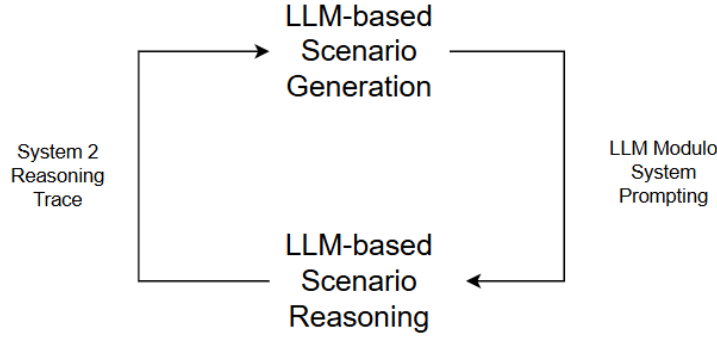


Figure 5.1: A circular relationship where effective scenario generation is needed for improving scenario reasoning and effective scenario reasoning is needed for improving scenario generation

Modulo Systems as conceptualized by Khambapati et. al are proposed to improve reasoning in LLMs. Here, symbolic planners are integrated with LLMs along with other tools such as verifiers to enhance reasoning. A similar, more generalized approach is proposed by Pan et. al in *LogicLM* and Kirtania et. al in *Logic-LM++* where a wide array of symbolic solvers such as solvers for constraint satisfaction problems, formal logic and more are utilized and their solutions to a given problem are added to the prompt of an LLM along with the corresponding question to have it generate an answer [104] [70]. Problem modeling by LLMs is the process of *LLM autoformalization* where natural language statements are transformed into formal specifications [150]. A consequence of using such symbolic approaches with LLMs along with the scenario definitions stated above results in a circular relationship where to improve scenario reasoning, we need to improve scenario generation and vice-versa. This is presented as a diagram in Figure 5.1.

In this chapter, I present an approach to generate scenarios using an architecture similar to an LLM Modulo System [88]. This architecture is presented in Figure 5.2. The LLM defines a real-world scenario as Planning Domain Definition Language (PDDL) *domain* and *problem* files for a *QA pair* from the Waymo

Reasoning dataset [80]. The former contains all the possible states and actions and the latter contains the initial state and the goal state for the planner. I use a basic breadth-first search based planner for creating the plan that addresses the problem. As a verification step, I have an LLM answer the initial question from the dataset based on all of this information. This is consistent with the approach used in LogicLM [104] where an LLM attempts to solve the given problem with the corresponding symbolic solver related files in its prompt. Consistent with the design of an LLM-modulo system, we also involve *syntax verifier VAL* in this approach. Both the PDDL files generated as well as the final attempt by an LLM are evaluated using an *LLM as a judge* [45]. Uniquely, the LLM as a judge carries out a comparative judgement with respect to the ground truth answer for the same question from the dataset. In this chapter, we are able to generate syntactically and semantically accurate scenarios for a diverse range of natural language descriptions from the Waymo Reasoning dataset.

The rest of this chapter is divided into sections as follows; Section 2 covers the method where design decisions for each component of the architecture are discussed. This includes the scenario generation as well as obtaining the planner output for the scenario and the various evaluations. Section 3 covers the evaluation along the lines of three of the four scenario generation challenges of focus in this dissertation: fidelity/realism, controllability and diversity. Finally, section 4 covers future works and discussions about how this project can be extended.

5.2 Method

The generation of the scenarios follows the design recommendations for an LLM Modulo System [88] where LLM based generation of symbolic representations are connected with verifiers. First, we preprocess the natural language QA data

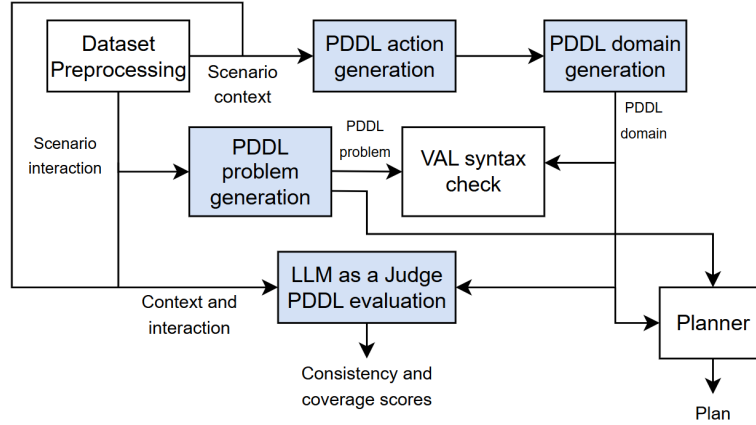


Figure 5.2: Model-based Task Planning Prompting of LLMs. The boxes in blue involve prompting an LLM and obtaining its output.

and separate the context and the vehicle interaction questions. The PDDL domain is then generated based on this context while the PDDL problem file is generated based on one of the vehicle interaction questions. This is followed by a syntax and semantic check by VAL and an LLM as a judge respectively. The domain and problem files are then given to a planner which generates a solution. Finally, for validation of the generated scenario, the PDDL domain, problem and plan file contents are given to an LLM to answer the initial question from the dataset. This answer undergoes a comparative evaluation by an LLM as a judge with respect to a reference answer from the dataset. Our dataset preprocessing script also has the capability to generate embeddings for the contexts mentioned above to evaluate diversity of the generated scenarios. The architecture diagram for this system is presented in 5.2. The rest of this section covers each of the components within this architecture.

5.2.1 Dataset and Preprocessing

The natural language based question and answer (QA) data I use is from the Waymo Open Motion Dataset - Reasoning (Waymo-Reasoning dataset) [80].

This dataset is based on the Waymo Open Motion dataset [38] which contains real-world trajectories. It contains real-world vehicle, pedestrian and cyclist trajectory data collected for the purpose of developing motion forecasting models. The Waymo-Reasoning dataset contains QA data pertaining to diverse interaction types such as ones based on traffic rules or human intentions. We preprocess this dataset to organize the subgroups *environment QA* and *surrounding agent QA* as a *scenario context* and the remaining subgroup *interaction QA* as the interaction QA. The evaluation task given to the language models is to answer the interaction questions given the scenario context. We considered having an LLM rewrite the context QA subgroup as a paragraph of text but since it contained inconsistencies with respect to the original data, we add the QA data as is into the context attribute. An exemplar parsed scenario is shown in Listing 5.3. Here, the file size is used as a proxy measure of scenario complexity since each scenario is contained in its own file. These file sizes are categorized as A, B and C representing small, medium and large file sizes respectively.

5.2.2 Prompting Using Model based Task Planning

We prompt an LLM to generate PDDL domain files for the given contextual information and a problem file for each scenario interaction in the parsed scenario data mentioned previously. We then perform a syntax verification of the LLM generated outputs using *VAL* [60]. In addition, we also carry out semantic verification of the generated PDDL files for coverage with respect to the facts in the given context. Once the generated PDDL file is syntax error free, we make use of a breadth first search based planner in the [74] library to create a plan that would solve the generated problem. The final prompt to the language model contains the scenario context, the interaction question, the PDDL domain model

that corresponds with the context, the PDDL problem that corresponds with the interaction question, and a plan generated by a symbolic solver that solves the aforementioned PDDL problem. We call this the *MBTP prompt*. This process is presented in Figure 5.2. An example of PDDL problem generation for the parsed scenario presented in Listing 5.3 is presented in 5.4. The semantic verification for this PDDL problem using LLM-as-a-Judge based evaluation is presented in Listing 5.5.

5.2.3 Response Evaluations using LLM as a Judge Approach

LLM as a Judge is used for evaluations of the generated PDDL files, reasoning of an LLM in response to the MBTP prompt explained in the previous subsection. Both of these assessments are *comparative judgements*. The coverage is with respect to the context information detailed above. The LLM answers are with respect to the corresponding ground truth answer for the question considered from the dataset. Both of these assessments result in a score out of 10 and an explanation generated by the LLM. The ideal generated scenario would adequately cover (a threshold score of 8 out of 10) the given information and enable an LLM to perform effective reasoning to answer a given question (a correctness score of 8 out of 10). This approach is used since the reference information is a single string of text and is significantly abstract.

We also evaluate the ground truth in the dataset itself using the LLM as a Judge approach mentioned above. This validates the reference answer from the dataset used for comparative evaluation above. In Listing 5.6, we can observe A ground truth evaluation of the scenario parsed in Listing 5.3 is presented in Listing 5.6. Both LLM as a Judge assessments are made using Deepseek V3. This

LLM was highly affordable and provided good capability in making judgements.

5.2.4 Diverse Scenario Generation

To ensure that generated scenarios are diverse, we make use of a clustering approach of a higher dimension representations of the scenario contexts from the dataset to determine scenario indices to generate. If the PDDL scenario generation scores detailed above are high for a representative set across clusters, we can conclude that the scenario generation is diverse. The clustering approach involves the transforming the context data mentioned above for a range of scenario indices into higher dimension embeddings based on the OpenAI embedding models text-embedding-3-small and text-embedding-3-large [100]. This is followed by applying the PCA algorithm for dimensionality reduction [79] followed by K-means clustering. An example of clustering carried for a set of scenarios is presented in Figure 5.3.

5.3 Evaluation

In chapter 1, four challenges regarding scenario generation are presented as the focus of this dissertation: (1) Fidelity/Realism of generated scenarios. (2) Controllability of the generator. (3) Diversity of the generated scenarios. (4) Transferability of the generated scenarios. The focus of the evaluation in this chapter is on the first three metrics. The latter metric can be assumed since the generation is in a well-known standardized format. It must be noted that this dataset contains natural language mappings of scenarios from the Waymo Motion Dataset [38] which contains real-world trajectory data. Therefore, for evaluating realism, the generated scenario will need to have a high degree of accuracy and

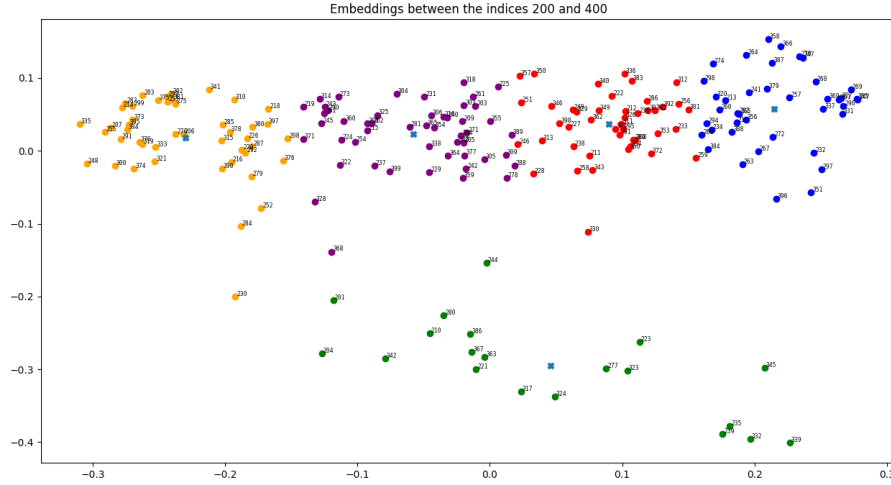


Figure 5.3: An example of clustering of scenario contexts for scenarios index range 200 to 400. Each point in the scatterplot represents a scenario context. The color of the point represents the cluster it belongs to with each cluster having a centroid represented by a blue cross mark. We can select a representative set of scenarios based on such a figure and assess diverse scenario generation capability of this system.

coverage with respect to the given question from the Waymo Reasoning dataset. For diversity we observe clusters of scenario *contexts* in this dataset and generate scenarios across a representative set from these clusters. Diversity is considered based on distance in this higher dimension space between scenario contexts.

In Listing 5.4, we can see the generated PDDL problem file for an interaction question for the preprocessed scenario. The PDDL problem file is based on a PDDL domain file generated based on the context information in the preprocessed data. Here, the ego vehicle and the surrounding agent #2 are represented as *vehicle0* and *vehicle2* respectively. We can observe the PDDL actions as well as the initial and goal states correspond to the facts in the context as well as the question in the preprocessed data respectively.

In Listing 5.5, we can observe an LLM as a judge score out of 10 about adequate coverage of the facts in the preprocessed data for the problem file generated above.

```

"Context": "Can you describe the layout of the lanes where the ego
agent is currently located? The ego agent is on the 1 lane from
the left , out of 3 lanes. What is the current velocity of the ego
agent? The ego agent's current speed is 12 m/s. What is the ego
agent's current action in terms of speed adjustment? The ego
agent is decelerating. Could you specify the ego agent's current
lane position? The ego agent is on the 1 lane from the left. What
type of agent is surrounding agent #0? Surrounding agent #0 is a
vehicle. How fast is surrounding agent #0 currently moving? Its
current speed is 16 m/s. What is the motion status of surrounding
agent #0? It is moving at a constant speed. Where is surrounding
agent #0 in relation to the ego agent? It is 7 meters in front
of the ego agent and 3 meters on the right. What is the type of
surrounding agent #2? Surrounding agent #2 is a vehicle. What is
the current speed of surrounding agent #2? Its current speed is
17 m/s. Can you describe the motion status of surrounding agent
#2? It is moving at a constant speed. Where is surrounding agent
#2 positioned relative to the ego agent? It is 19 meters in front
of the ego agent and 3 meters on the right.",
...
"Interactions_1": { "reference_question": "How will the ego agent and
surrounding agent #2 interact on the road?",
"reference_answer": "Surrounding agent #2 will maintain its
position ahead of the ego agent as both are moving in the
same direction with surrounding agent #2 at a constant speed
and the ego agent decelerating." ...

```

Listing 5.3: Example parsed scenario after preprocessing. Few details have been omitted for conciseness.

```

(define (problem ego-vehicle-scenario-v2)
  ...
  (: objects
    ego - agent
    lane1 lane2 lane3 - lane
    vehicle0 vehicle2 - vehicle
  )
  (: init
    ...
    ; Vehicle0: 7m ahead, 3m right (front-right), in lane2
    (vehicle-at-fast-speed vehicle0)
    (vehicle-at-constant-speed vehicle0)
    (vehicle-is-in-lane vehicle0 lane2)
    (vehicle-is-in-front-right-of-ego vehicle0)
    (ego-near-vehicle-front-right vehicle0)
    (vehicle-in-front-right vehicle0)

    ; Vehicle2: 19m ahead, 3m right (far front-right), in lane2
    (vehicle-at-fast-speed vehicle2)
    (vehicle-at-constant-speed vehicle2)
    (vehicle-is-in-lane vehicle2 lane2)
    (vehicle-is-far-ahead-right-of-ego vehicle2)
    (vehicle-far-in-front-right vehicle2)
  )
  (: goal
    (ego-is-decelerating)
  )
)

```

Listing 5.4: PDDL problem generated by the LLM GPT 4.1 for the interaction question within the scenario parsed in Listing 5.3 and after the PDDL domain is generated. We can observe the states in this file correspond to the parsed interaction question and answer. Few details have been omitted for conciseness.

```

"Problem data provided": "How will the ego agent and surrounding
agent #2 interact on the road?",
"Score explanation": "The problem file directly models the
interaction context: it places the ego in the leftmost lane, next
to the two relevant vehicles, with all spatial and speed
predicates correctly instantiated for vehicle0 (close, front-
right) and vehicle2 (farther front-right). The goal is set for
the ego to be decelerating, reflecting a cautious or responsive
behavior to the nearby (closer) vehicle in the adjacent lane.
While the explicit interaction with agent #2 (vehicle2) is less
emphasized in the goal than with vehicle0, the presence and
predicates for vehicle2 show that its position is accounted for
in the conditions surrounding the ego's behavior. A minor
shortcoming may be the lack of an explicit future state following
the interaction (such as passing or merging), but for the
deceleration-focused goal and the state of the road given, the
coverage is very strong.",
"Grade": "9", ...

```

Listing 5.5: PDDL Problem Coverage Evaluation using GPT4.1 LLM as a judge for the problem presented in Listing 5.4

5.4 Conclusion

In this chapter, we presented Cruzway Scenario Generator, an approach to generate scenarios using an LLM in the Planning Domain Definition Language (PDDL) format. The scenarios are generated based on a natural language description of scenarios and vehicle interaction questions from the Waymo Open Motion Dataset - Reasoning, a dataset based on the Waymo Open Motion Dataset which contains real-world trajectory data for training motion forecasting models. The generated scenarios undergo syntax and semantic verification as well as a reasoning validation based on an LLM prompting approach suggested in LogicLM [104]. In addition, the natural language scenario data is clustered to assess diverse scenario generation capabilities in Cruzway Scenario Generator. With this

```
"Given Question": "How will the ego agent and surrounding agent #2  
interact on the road?",  
"Ground Truth Answer": "Surrounding agent #2 will maintain its  
position ahead of the ego agent as both are moving in the same  
direction with surrounding agent #2 at a constant speed and the  
ego agent decelerating.",  
"Ground Truth Grades": {"Correctness score": "8",  
"Correctness explanation": "The answer is mostly correct but lacks  
precision. Surrounding agent #2 is indeed maintaining its  
position ahead of the ego agent, as it is moving at a constant  
speed of 17 m/s, which is faster than the ego agent's speed of 12  
m/s (while decelerating). However, the answer could be improved  
by explicitly mentioning the speed difference and the fact that  
the distance between them will increase over time due to this  
speed difference.",  
"Updated Answer": "Surrounding agent #2 will maintain its position  
ahead of the ego agent and increase the distance over time, as it  
is moving at a constant speed of 17 m/s, which is faster than  
the ego agent's speed of 12 m/s while decelerating.", ...
```

Listing 5.6: Ground truth evaluation using LLM as a judge for the scenario parsed in Listing 5.3. Few details have been omitted for conciseness.

system, we are able to generate realistic and diverse scenarios in a controllable and interpretable manner.

Bibliography

- [1] Erfan Aasi, Phat Nguyen, Shiva Sreeram, Guy Rosman, Sertac Karaman, and Daniela Rus. Generating out-of-distribution scenarios using language models. *arXiv preprint arXiv:2411.16554*, 2024.
- [2] Hassane Abouaissa, Daniel Jolly, Amar Benasser, et al. Macro-micro simulation of traffic flow. *IFAC Proceedings Volumes*, 39(3):351–356, 2006.
- [3] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [4] Syeda Nahida Akter, Aman Madaan, Sangwu Lee, Yiming Yang, and Eric Nyberg. Self-imagine: Effective unimodal reasoning with multimodal models using self-imagination. *arXiv preprint arXiv:2401.08025*, 2024.
- [5] 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. IEEE, 2017.
- [6] Shyamal Anadkat. How to make your completions outputs consistent with the new seed parameter. https://cookbook.openai.com/examples/reproducible_outputs_with_the_seed_parameter, November 2023.
- [7] Mikael Andersson, Irene Natale, Andreas Tingberg, and Jacob Kath. Pyoscx scenariogeneration. <https://github.com/pyoscx/scenariogeneration>, 2025. Github Repository, accessed March 18 2025.
- [8] ApolloAuto. Apollo Dreamland. https://github.com/ApolloAuto/apollo/blob/master/docs/specs/Dreamland_introduction.md, 2019. Web page, accessed 12 March 2020.
- [9] P. Armstrong. Bloom’s taxonomy. <https://cft.vanderbilt.edu/guides-sub-pages/blooms-taxonomy/>, 2010. Web page, accessed May 01, 2024.

- [10] ASAM. Asam openxontology. <https://www.asam.net/project-detail/asam-openxontology/>, 2024. Web page, accessed 22 April 2024.
- [11] ASAM. Asam openscenario. <https://www.asam.net/index.php?eID=dumpFile&t=f&f=4092&token=d3b6a55e911b22179e3c0895fe2caae8f5492467>, 2025. Web page, accessed 30 January 2025.
- [12] Autoware. Autoware.AI. <https://www.autoware.ai/>, 2020. Web page, accessed 12 March 2020.
- [13] Gerrit Bagschik, Till Menzel, and Markus Maurer. Ontology based scene creation for the development of automated vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1813–1820. IEEE, 2018.
- [14] Inc. Baidu. Baidu apollo github repository. <https://github.com/ApolloAuto/apollo>, 2025. Accessed 13 April 2025.
- [15] Yuanhao Ban, Ruochen Wang, Tianyi Zhou, Minhao Cheng, Boqing Gong, and Cho-Jui Hsieh. Understanding the impact of negative prompts: When and how do they take effect? In *European Conference on Computer Vision*, pages 190–206. Springer, 2024.
- [16] Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*, 2023.
- [17] UC Berkeley. Transportation injury mapping system. <https://tims.berkeley.edu/help/SWITRS.php>, 2024. Web page, accessed 17 April 2024.
- [18] David Burton, Amanda Delaney, Stuart Newstead, David Logan, and Brian Fildes. Evaluation of anti-lock braking systems effectiveness. *Accident Analysis and Prevention*, 29(6):745–757, 1997.
- [19] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.
- [20] CARLA. Carla 0.9.1: Vehicle navigation, new waypoint-based api, maps creation and more. <http://carla.org/2018/11/16/release-0.9.1/>. Web page, accessed 12 March 2020.

- [21] CARLA. Carla 0.9.8 release. <http://carla.org/2020/03/09/release-0.9.8/>. Web page, accessed 14 May 2020.
- [22] CARLA. ScenarioRunner for CARLA. https://github.com/carla-simulator/scenario_runner. Web page, accessed 12 March 2020.
- [23] CARLA. Carla scenario runner. https://github.com/carla-simulator/scenario_runner, 2024. Github Repository, accessed March 18 2025.
- [24] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8748–8757, 2019.
- [25] Li Chen, Penghao Wu, Kashyap Chitta, Bernhard Jaeger, Andreas Geiger, and Hongyang Li. End-to-end autonomous driving: Challenges and frontiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [26] Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.
- [27] Michele Colledanchise and Petter Ögren. *Behavior Trees in Robotics and AI: An Introduction*. CRC Press, 2019.
- [28] Can Cui, Yunsheng Ma, Xu Cao, Wenqian Ye, Yang Zhou, Kaizhao Liang, Jintai Chen, Juanwu Lu, Zichong Yang, Kuei-Da Liao, et al. A survey on multimodal large language models for autonomous driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 958–979, 2024.
- [29] Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. Explaining answers with entailment trees. *arXiv preprint arXiv:2104.08661*, 2021.
- [30] Matthew DeBord. Waymo just crossed 10 million self-driving miles – but the company has a secret weapon that gives it even more of an edge. <https://www.businessinsider.com/waymo-self-driving-cars-secret-weapon-is-simulation-testing-2018-10>, 2018. Web page, accessed 17 April 2019.
- [31] Deepdrive Voyage. Build self-driving ai. <https://deepdrive.voyage.auto/>. Web page, accessed 12 March 2020.

- [32] Yao Deng, Jiaohong Yao, Zhi Tu, Xi Zheng, Mengshi Zhang, and Tianyi Zhang. Target: Automated scenario generation from traffic rules for testing autonomous vehicles. *arXiv preprint arXiv:2305.06018*, 2023.
- [33] 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.
- [34] 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.
- [35] 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.
- [36] Jacob Doughty, Zipiao Wan, Anishka Bompelli, Jubahed Qayum, Taozhi Wang, Juran Zhang, Yujia Zheng, Aidan Doyle, Pragnya Sridhar, Arav Agarwal, Christopher Bogart, Eric Keylor, Can Kultur, Jaromir Savelka, and Majd Sakr. A comparative study of ai-generated (gpt-4) and human-crafted mcqs in programming education. In *Proceedings of the 26th Australasian Computing Education Conference, ACE 2024*. ACM, January 2024.
- [37] Karim Elmaaroufi, Devan Shanker, Ana Cismaru, Marcell Vazquez-Chanlatte, Alberto Sangiovanni-Vincentelli, Matei Zaharia, and Sanjit A Seshia. Scenicnl: generating probabilistic scenario programs from natural language. *arXiv preprint arXiv:2405.03709*, 2024.
- [38] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9710–9719, 2021.
- [39] Lan Feng, Quanyi Li, Zhenghao Peng, Shuhan Tan, and Bolei Zhou. Trafficgen: Learning to generate diverse and realistic traffic scenarios. In *2023 IEEE international conference on robotics and automation (ICRA)*, pages 3567–3575. IEEE, 2023.
- [40] Alexander J Fiannaca, Chinmay Kulkarni, Carrie J Cai, and Michael Terry. Programming without a programming language: Challenges and opportunities for designing developer tools for prompt programming. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–7, 2023.

- [41] Giorgio Franceschelli and Mirco Musolesi. On the creativity of large language models. *arXiv preprint arXiv:2304.00008*, 2023.
- [42] Daniel J Fremont, Tommaso Dreossi, Shromona Ghosh, Xiangyu Yue, Alberto L Sangiovanni-Vincentelli, and Sanjit A Seshia. Scenic: a language for scenario specification and scene generation. In *Proceedings of the 40th ACM SIGPLAN conference on programming language design and implementation*, pages 63–78, 2019.
- [43] Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*, 2023.
- [44] Alessio Gambi, Tri Huynh, and Gordon Fraser. Generating effective test cases for self-driving cars from police reports. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 257–267, 2019.
- [45] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024.
- [46] Yu Gu, Boyuan Zheng, Boyu Gou, Kai Zhang, Cheng Chang, Sanjari Srivastava, Yanan Xie, Peng Qi, Huan Sun, and Yu Su. Is your llm secretly a world model of the internet? model-based planning for web agents. *arXiv preprint arXiv:2411.06559*, 2024.
- [47] Lin Guan, Karthik Valmeekam, Sarath Sreedharan, and Subbarao Kambhampati. Leveraging pre-trained large language models to construct and utilize world models for model-based task planning. *Advances in Neural Information Processing Systems*, 36:79081–79094, 2023.
- [48] Cole Gulino, Justin Fu, Wenjie Luo, George Tucker, Eli Bronstein, Yiren Lu, Jean Harb, Xinlei Pan, Yan Wang, Xiangyu Chen, et al. Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research. *Advances in Neural Information Processing Systems*, 36:7730–7742, 2023.
- [49] An Guo, Yuan Zhou, Haoxiang Tian, Chunrong Fang, Yunjian Sun, Weisong Sun, Xinyu Gao, Anh Tuan Luu, Yang Liu, and Zhenyu Chen. Sovar: Build generalizable scenarios from accident reports for autonomous driving testing. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, pages 268–280, 2024.

- [50] Matthew Guzdial, Sam Snodgrass, and Adam J Summerville. Classical pcg. In *Procedural Content Generation via Machine Learning: An Overview*, pages 7–22. Springer, 2022.
- [51] Niklas Hanselmann, Katrin Renz, Kashyap Chitta, Apratim Bhattacharyya, and Andreas Geiger. King: Generating safety-critical driving scenarios for robust imitation via kinematics gradients. In *European Conference on Computer Vision*, pages 335–352. Springer, 2022.
- [52] Shibo Hao, Yi Gu, Haotian Luo, Tianyang Liu, Xiyan Shao, Xinyuan Wang, Shuhua Xie, Haodi Ma, Adithya Samavedhi, Qiyue Gao, et al. Llm reasoners: New evaluation, library, and analysis of step-by-step reasoning with large language models. *arXiv preprint arXiv:2404.05221*, 2024.
- [53] Andrew J. Hawkings. It’s been a rough year for robotaxis — but not for waymo. <https://www.theverge.com/24323593/waymo-year-in-review-robotaxi-trips-miles-growth-cost>, 2024. Web page, accessed 27 April 2025.
- [54] Andrew J. Hawkings. Waymo issues software and mapping recall after robotaxi crashes into a telephone pole. <https://www.theverge.com/2024/6/12/24175489/waymo-recall-telephone-poll-crash-phoenix-software-map>, 2024. Web page, accessed 6 July 2024.
- [55] Andrew J. Hawkings. Waymo’s robotaxis are under investigation for crashes and traffic law violations. <https://www.theverge.com/2024/5/14/24156238/waymo-nhtsa-investigation-crash-wrong-side-road>, 2024. Web page, accessed 6 July 2024.
- [56] Andrew J. Hawkings. Welcome to simulation city, the virtual world where waymo tests its autonomous vehicles. <https://www.theverge.com/2021/7/6/22565448/waymo-simulation-city-autonomous-vehicle-testing-virtual>, 2024. Web page, accessed 9 July 2024.
- [57] Andrew J. Hawkins. Waymo is now offering 24/7 robotaxi rides in silicon valley. <https://www.theverge.com/news/627619/waymo-silicon-valley-robotaxi-bay-area-service-area>, 2025. Accessed 13 April 2025.
- [58] Andrew J. Hawkins. Waymo readies autonomous cars for first international tests in japan. <https://www.theverge.com/news/645777/waymo-japan-autonomous-test-robotaxi-international>, 2025. Accessed 13 April 2025.

- [59] Joey Hong, Anca Dragan, and Sergey Levine. Q-sft: Q-learning for language models via supervised fine-tuning. *arXiv preprint arXiv:2411.05193*, 2024.
- [60] Richard Howey, Derek Long, and Maria Fox. Val: Automatic plan validation, continuous effects and mixed initiative planning using pddl. In *16th IEEE International Conference on Tools with Artificial Intelligence*, pages 294–301. IEEE, 2004.
- [61] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55, 2025.
- [62] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.
- [63] Ishaan Jaff et al. Litellm. <https://github.com/BerriAI/litellm>, 2024. Web page, accessed May 01, 2024.
- [64] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.
- [65] Daniel Kahneman. Thinking, fast and slow. *Farrar, Straus and Giroux*, 2011.
- [66] Nidhi Kalra and Susan M Paddock. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation Research Part A: Policy and Practice*, 94:182–193, 2016.
- [67] Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. Llms can’t plan, but can help planning in llm-modulo frameworks. *arXiv preprint arXiv:2402.01817*, 2024.
- [68] Zixuan Ke, Fangkai Jiao, Yifei Ming, Xuan-Phi Nguyen, Austin Xu, Do Xuan Long, Minzhi Li, Chengwei Qin, Peifeng Wang, Silvio Savarese, et al. A survey of frontiers in llm reasoning: Inference scaling, learning to reason, and agentic systems. *arXiv preprint arXiv:2504.09037*, 2025.
- [69] Zachary Kenton, Tom Everitt, Laura Weidinger, Iason Gabriel, Vladimir Mikulik, and Geoffrey Irving. Alignment of language agents. *arXiv preprint arXiv:2103.14659*, 2021.

- [70] Shashank Kirtania, Priyanshu Gupta, and Arjun Radhakirshna. Logic-lm++: Multi-step refinement for symbolic formulations. *arXiv preprint arXiv:2407.02514*, 2024.
- [71] Moritz Klischat, Edmond Irani Liu, Fabian Holtke, and Matthias Althoff. Scenario factory: Creating safety-critical traffic scenarios for automated vehicles. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–7. IEEE, 2020.
- [72] Philip Koopman. Sae j3016 user guide. <https://users.ece.cmu.edu/~koopman/j3016/index.html>, 2025. Web page, accessed 24 May 2025.
- [73] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.
- [74] Erwin Lejeune. Jupyddl, an extensible python library for pddl planning and parsing, 2021.
- [75] LGSVL. LGSVL simulator: An autonomous vehicle simulator. <https://github.com/lgsvl/simulator>. Web page, accessed 12 March 2020.
- [76] LGSVL Simulator. LGSVL simulator overview. <https://www.lgsvlsimulator.com/downloads/LGSVL-Simulator-Overview.pdf>, 2020. Web page, accessed 10 March 2020.
- [77] Ao Li, Shitao Chen, Liting Sun, Nanning Zheng, Masayoshi Tomizuka, and Wei Zhan. Scegene: Bio-inspired traffic scenario generation for autonomous driving testing. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):14859–14874, 2021.
- [78] Johnny Li, Saksham Consul, Eda Zhou, James Wong, Naila Farooqui, Yuxin Ye, Nithyashree Manohar, Zhuxiaona Wei, Tian Wu, Ben Echols, et al. Banishing llm hallucinations requires rethinking generalization. *arXiv preprint arXiv:2406.17642*, 2024.
- [79] Lingjun Li, Shigang Liu, Yali Peng, and Zengguo Sun. Overview of principal component analysis algorithm. *Optik*, 127(9):3935–3944, 2016.
- [80] Yiheng Li, Chongjian Ge, Chenran Li, Chenfeng Xu, Masayoshi Tomizuka, Chen Tang, Mingyu Ding, and Wei Zhan. Womd-reasoning: A large-scale language dataset for interaction and driving intentions reasoning. *arXiv preprint arXiv:2407.04281*, 2024.

- [81] Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, et al. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025.
- [82] Fang Liu, Yang Liu, Lin Shi, Houkun Huang, Ruifeng Wang, Zhen Yang, and Li Zhang. Exploring and evaluating hallucinations in llm-powered code generation. *arXiv preprint arXiv:2404.00971*, 2024.
- [83] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*, 2023.
- [84] 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.
- [85] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.
- [86] Scott Lundberg et al. Guidance. <https://github.com/guidance-ai/guidance>, 2024. Web page, accessed May 01, 2024.
- [87] Alexis C. Madrigal. Inside Waymo’s secret world for training self-driving cars. <https://www.theatlantic.com/technology/archive/2017/08/inside-waymos-secret-testing-and-simulation-facilities/537648/>, 2018. Web page, accessed 17 April 2019.
- [88] Sadegh Mahdavi, Raquel Aoki, Keyi Tang, and Yanshuai Cao. Leveraging environment interaction for automated pddl translation and planning with large language models. *arXiv preprint arXiv:2407.12979*, 2024.
- [89] Osama Makansi, Özgün Çiçek, Yassine Marrakchi, and Thomas Brox. On exposing the challenging long tail in future prediction of traffic actors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13147–13157, 2021.
- [90] MATLAB. Matlab roadrunner. <https://www.mathworks.com/products/roadrunner.html>, 2025. Product Webpage, accessed March 18 2025.

- [91] Drew M McDermott. The 1998 ai planning systems competition. *AI magazine*, 21(2):35–35, 2000.
- [92] Mechanical Simulation Corporation. CarSIM. <https://www.carsim.com/>, 2020. Web page, accessed 12 March 2020.
- [93] Till Menzel, Gerrit Bagschik, Leon Isensee, Andre Schomburg, and Markus Maurer. From functional to logical scenarios: Detailing a keyword-based scenario description for execution in a simulation environment. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 2383–2390. IEEE, 2019.
- [94] Microsoft. Airsim. <https://microsoft.github.io/AirSim/>, 2020. Web page, accessed 12 March 2020.
- [95] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey. *arXiv preprint arXiv:2402.06196*, 2024.
- [96] Roshanak Mirzaee, Hossein Rajaby Faghihi, Qiang Ning, and Parisa Kordjmeshidi. Spartqa:: A textual question answering benchmark for spatial reasoning. *arXiv preprint arXiv:2104.05832*, 2021.
- [97] National Transportation Safety Board. Collision between a sport utility vehicle operating with partial driving automation and a crash attenuator. <https://www.nts.gov/news/events/Documents/2020-HWY18FH011-BMG-abstract.pdf>. Meeting minutes, Feb. 25, 2020.
- [98] Phat Nguyen, Tsun-Hsuan Wang, Zhang-Wei Hong, Sertac Karaman, and Daniela Rus. Text-to-drive: Diverse driving behavior synthesis via large language models. *arXiv preprint arXiv:2406.04300*, 2024.
- [99] NVIDIA. NVIDIA Drive - autonomous vehicle development platforms. <https://developer.nvidia.com/drive>, 2020. Web page, accessed 10 March 2020.
- [100] OpenAI. Vector embeddings (documentation). <https://platform.openai.com/docs/guides/embeddings/embedding-models>.
- [101] OpenODD. Asam openodd base standard 1.0.0. https://publications.pages.asam.net/standards/ASAM_OpenODD/ASAM_OpenODD/latest/specification/index.html, 2025. Web page, accessed 5 May 2025.
- [102] OpenStreetMap contributors. Planet dump retrieved from <https://planet.osm.org>. <https://www.openstreetmap.org>, 2017.

- [103] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
- [104] Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logiclm: Empowering large language models with symbolic solvers for faithful logical reasoning. *arXiv preprint arXiv:2305.12295*, 2023.
- [105] Ishaan Paranjape. IntGen. <https://github.com/AugmentedDesignLab/intgen>, 2019. Web page, accessed 12 March 2020.
- [106] Ishaan Paranjape. Modified Netgenerate. <https://github.com/AugmentedDesignLab/sumo-mirror/tree/sumoIntgen>, 2019. Web page, accessed 14 May 2020.
- [107] Ishaan Paranjape. Sumo2Unreal. <https://github.com/AugmentedDesignLab/Sumo2Unreal>, 2019. Web page, accessed 12 March 2020.
- [108] Ishaan Paranjape. blenderopendriveimporter: Generate and import opendrive roads into blender. <https://github.com/ishaan95/blenderOpenDRIVEImporter>, 2023. Github Repository, accessed March 17 2025.
- [109] Ishaan Paranjape, Abdul Jawad, Yanwen Xu, Asiih Song, and Jim Whitehead. A modular architecture for procedural generation of towns, intersections and scenarios for testing autonomous vehicles. In *2020 ieee intelligent vehicles symposium (iv)*, pages 162–168. IEEE, 2020.
- [110] Moschoula Pternea, Prerna Singh, Abir Chakraborty, Yagna Oruganti, Mirco Milletari, Sayli Bapat, and Kebei Jiang. The rl/llm taxonomy tree: Reviewing synergies between reinforcement learning and large language models. *arXiv preprint arXiv:2402.01874*, 2024.
- [111] Tianwen Qian, Jingjing Chen, Linhai Zhuo, Yang Jiao, and Yu-Gang Jiang. Nuscenes-qa: A multi-modal visual question answering benchmark for autonomous driving scenario. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 4542–4550, 2024.
- [112] Lianhui Qin, Aditya Gupta, Shyam Upadhyay, Luheng He, Yejin Choi, and Manaal Faruqi. Timedial: Temporal commonsense reasoning in dialog. *arXiv preprint arXiv:2106.04571*, 2021.

- [113] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [114] rFpro. rfpro: Simulation software. <http://www.rfpro.com/about/>, 2020. Web page, accessed 12 March 2020.
- [115] 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.
- [116] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, et al. Lgsvl simulator: A high fidelity simulator for autonomous driving. *arXiv preprint arXiv:2005.03778*, 2020.
- [117] Luke Rowe, Roger Girgis, Anthony Gosselin, Liam Paull, Christopher Pal, and Felix Heide. Scenario dreamer: Vectorized latent diffusion for generating driving simulation environments. *arXiv preprint arXiv:2503.22496*, 2025.
- [118] Christina M Rudin-Brown and Heather A Parker. Behavioural adaptation to adaptive cruise control (acc): implications for preventive strategies. *Transportation Research Part F: Traffic Psychology and Behaviour*, 7(2):59–76, 2004.
- [119] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. A systematic survey of prompt engineering in large language models: Techniques and applications. *arXiv preprint arXiv:2402.07927*, 2024.
- [120] Johannes Schmitz. Blender driving scenario creator. <https://github.com/johschmitz/blender-driving-scenario-creator>, 2025. Github Repository, accessed March 18 2025.
- [121] Maike Scholtes, Lukas Westhofen, Lara Ruth Turner, Katrin Lotto, Michael Schuldes, Hendrik Weber, Nicolas Wagener, Christian Neurohr, Martin Herbert Bollmann, Franziska Körtke, et al. 6-layer model for a structured description and categorization of urban traffic and environment. *IEEE Access*, 9:59131–59147, 2021.
- [122] Barbara Schütt, Joshua Ransiek, Thilo Braun, and Eric Sax. 1001 ways of scenario generation for testing of self-driving cars: A survey. In *2023 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–8. IEEE, 2023.

- [123] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering. In *European Conference on Computer Vision*, pages 256–274. Springer, 2025.
- [124] Pavel Smirnov, Frank Joublin, Antonello Ceravola, and Michael Gienger. Generating consistent pddl domains with large language models. *arXiv preprint arXiv:2404.07751*, 2024.
- [125] Gillian Smith. Understanding procedural content generation: a design-centric analysis of the role of pcg in games. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 917–926, 2014.
- [126] Jessie Smith and Viktor Lundqvist. Scaling simulation. <https://blog.aurora.tech/engineering/scaling-simulation>, 2024. Web page, accessed 9 July 2024.
- [127] Asiih Song and Jim Whitehead. Townsim: agent-based city evolution for naturalistic road network generation. In *Proc. 10th Workshop on Procedural Content Generation (PCG 2019)*, pages 1–9, 2019.
- [128] Sebastian Söntges and Matthias Althoff. Computing the drivable area of autonomous road vehicles in dynamic road scenes. *IEEE Transactions on Intelligent Transportation Systems*, 19(6):1855–1866, 2017.
- [129] Shiva Sreeram, Tsun-Hsuan Wang, Alaa Maalouf, Guy Rosman, Sertac Karaman, and Daniela Rus. Probing multimodal llms as world models for driving. *arXiv preprint arXiv:2405.05956*, 2024.
- [130] Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. Table meets llm: Can large language models understand structured table data? a benchmark and empirical study. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 645–654, 2024.
- [131] Adam Summerville, Sam Snodgrass, Matthew Guzdial, Christoffer Holmgård, Amy K Hoover, Aaron Isaksen, Andy Nealen, and Julian Togelius. Procedural content generation via machine learning (pcgml). *IEEE Transactions on Games*, 10(3):257–270, 2018.
- [132] Chen Sun, Zejian Deng, Wenbo Chu, Shen Li, and Dongpu Cao. Acclimatizing the operational design domain for autonomous driving systems. *IEEE Intelligent Transportation Systems Magazine*, 14(2):10–24, 2021.

- [133] Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. Traffic-sim: Learning to simulate realistic multi-agent behaviors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10400–10409, 2021.
- [134] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*, 2018.
- [135] Shuncheng Tang, Zhenya Zhang, Jixiang Zhou, Lei Lei, Yuan Zhou, and Yinxing Xue. Legend: A top-down approach to scenario generation of autonomous driving systems assisted by large language models. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, pages 1497–1508, 2024.
- [136] Yun Tang, Antonio A Bruto Da Costa, Xizhe Zhang, Irvine Patrick, Siddhartha Khastgir, and Paul Jennings. Domain knowledge distillation from large language model: An empirical study in the autonomous driving domain. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 3893–3900. IEEE, 2023.
- [137] TASS International. Prescan: Simulation of adas & active safety. <https://tass.plm.automation.siemens.com/prescan>, 2020. Web page, accessed 12 March 2020.
- [138] Eric Thorn, Shawn C Kimmel, Michelle Chaka, et al. A framework for automated driving system testable cases and scenarios. Technical report, United States. Department of Transportation. National Highway Traffic Safety . . . , 2018.
- [139] Simon Ulbrich, Till Menzel, Andreas Reschka, Fabian Schuldt, and Markus Maurer. Defining and substantiating the terms scene, situation, and scenario for automated driving. In *2015 IEEE 18th international conference on intelligent transportation systems*, pages 982–988. IEEE, 2015.
- [140] Unreal Engine. Behavior tree overview. <https://docs.unrealengine.com/en-US/Engine/ArtificialIntelligence/BehaviorTrees/BehaviorTreesOverview/index.html>, 2020. Web page, accessed 12 March 2020.
- [141] Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models-a critical investigation. *Advances in Neural Information Processing Systems*, 36:75993–76005, 2023.

- [142] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [143] Song Wang and Zhixia Li. Exploring causes and effects of automated vehicle disengagement using statistical modeling and classification tree based on field test data. *Accident Analysis & Prevention*, 129:44–54, 2019.
- [144] Irene Weber. Large language models as software components: A taxonomy for llm-integrated applications. *arXiv preprint arXiv:2406.10300*, 2024.
- [145] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.
- [146] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.
- [147] Lukas Westhofen, Christian Neurohr, Tjark Koopmann, Martin Butz, Barbara Schütt, Fabian Utesch, Birte Neurohr, Christian Gutenkunst, and Eckard Böde. Criticality metrics for automated driving: A review and suitability analysis of the state of the art. *Archives of Computational Methods in Engineering*, 30(1):1–35, 2023.
- [148] Kyle Wiggers. Aurora urges autonomous vehicle industry to adopt better safety metrics. <https://venturebeat.com/2020/01/24/aurora-urges-autonomous-vehicle-industry-to-adopt-better-safety-metrics/>, 2020. Web page, accessed 9 March 2020.
- [149] Wikipedia. Emergence. <https://en.wikipedia.org/wiki/Emergence>, 2020. Web page, accessed 12 March 2020.
- [150] Yuhuai Wu, Albert Qiaochu Jiang, Wenda Li, Markus Rabe, Charles Staats, Mateja Jamnik, and Christian Szegedy. Autoformalization with large language models. *Advances in Neural Information Processing Systems*, 35:32353–32368, 2022.
- [151] Lan Yang, Yipeng Yang, Guoyuan Wu, Xiangmo Zhao, Shan Fang, Xishun Liao, Runmin Wang, and Mengxiao Zhang. A systematic review of autonomous emergency braking system: impact factor, technology, and performance evaluation. *Journal of advanced transportation*, 2022(1):1188089, 2022.

- [152] Zhenjie Yang, Xiaosong Jia, Hongyang Li, and Junchi Yan. Llm4drive: A survey of large language models for autonomous driving. *arXiv preprint arXiv:2311.01043*, 2023.
- [153] Ilker Yildirim and LA Paul. From task structures to world models: what do llms know? *Trends in Cognitive Sciences*, 2024.
- [154] Wei Zhan, Liting Sun, Di Wang, Haojie Shi, Aubrey Clausse, Maximilian Naumann, Julius Kummerle, Hendrik Konigshof, Christoph Stiller, Arnaud de La Fortelle, et al. Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps. *arXiv preprint arXiv:1910.03088*, 2019.
- [155] Chenshuang Zhang, Chaoning Zhang, Mengchun Zhang, and In So Kweon. Text-to-image diffusion models in generative ai: A survey. *arXiv preprint arXiv:2303.07909*, 2023.
- [156] Jiawei Zhang, Chejian Xu, and Bo Li. Chatscene: Knowledge-enabled safety-critical scenario generation for autonomous vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15459–15469, 2024.
- [157] Yongqi Zhao, Ji Zhou, Dong Bi, Tomislav Mihalj, Jia Hu, and Arno Eichberger. A survey on the application of large language models in scenario-based testing of automated driving systems. *arXiv preprint arXiv:2505.16587*, 2025.
- [158] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36, 2024.
- [159] Yupeng Zheng, Zebin Xing, Qichao Zhang, Bu Jin, Pengfei Li, Yuhang Zheng, Zhongpu Xia, Kun Zhan, Xianpeng Lang, Yaran Chen, et al. Plan-agent: A multi-modal large language agent for closed-loop vehicle motion planning. *arXiv preprint arXiv:2406.01587*, 2024.
- [160] Chong Zhou, Chen Change Loy, and Bo Dai. Extract free dense labels from clip. In *European Conference on Computer Vision*, pages 696–712. Springer, 2022.