Text2Scenario: Text-Driven Scenario Generation for Autonomous Driving Test

Xuan Cai[1], Zhiyong Cui[1*], Xuesong Bai[1], Danmu Xie[1], Haiyang Yu[1,2], Yilong Ren[1,2*]

[1*]State Key Laboratory of Intelligent Transportation Systems, School of Transportation Science and Technology, Beihang University, No. 9 South Third Street, Beijing, 100191, Beijing, P.R.China.
[2*]Zhongguancun National Laboratory, Beijing, 100191, Beijing, P.R.China.

*Corresponding author(s). E-mail(s): zhiyongc@buaa.edu.com; yilongren@buaa.edu.com;
Contributing authors: caixuan@buaa.edu.com; xs_bai@buaa.edu.com; xiedanmu@163.com; hyyu@buaa.edu.com;

Abstract

Autonomous driving (AD) testing represents a vital tool for validating performance levels prior to the release of products. The development of sliced scenarios within a simulated environment is recognized as both a secure and efficacious strategy, yet the process of its customization frequently entails laborious, intricate, and protracted manual undertakings, which in turn decelerates the progress and deployment of AD technologies. In light of this, we introduce the Text2Scenario, a framework that harnesses a Large Language Model (LLM) to autonomously generate simulation test scenarios with a high degree of match, derived from the user's natural language input. Specifically, an LLM equipped with a meticulously engineered input prompt scheme functions as a text parser for test scenario descriptions, selecting from a hierarchically organized scenario repository those elements that reflect the user's preference. Subsequently, leveraging the precedence of the scenario components, the process involves sequentially matching and linking scenario representations within a Domain Specific Language corpus, thus fabricating executable test scenarios. The experimental outcomes demonstrate that such prompt engineering can scrupulously draw out the subtle details of scenario elements embedded within various descriptive types, where the vast majority of the crafted scenarios comply with the user's initial expectations,

allowing for the efficient and precise evaluation of diverse AD stacks void of the cumbersome requirement for manual scenario configuration.

Keywords: Scenario Generation, Large Language Model, Autonomous Driving Test, Domain Specific Language

Abbreviations

AD Autonomous Driving
ADAS Advanced Driver Assistance System
ADS Autonomous Driving System
C-NCAP China New Car Assessment Program
CoT Chain of Thought
DSL Domain Specific Language
FS Few-Shot
LLM Large Language Model
NHTSA National Highway Traffic Safety Administration
RQ Research Question
SAC Syntax Alignment Checking
SC Self-Consistency
SOTIF Safety of The Intended Functionality
SUT System Under Test
SysML Systems Modeling Language
T2S Text to Scenario
UML Unified Modeling Language

1 Introduction

The emergence of AD technology heralds a transformative era with the promise of enhancing traffic safety and mobility efficiency [Yan+19][PWL21]. Prior to its full-scale commercial adoption, the technology must be subjected to stringent evaluations to affirm its dependability and safety credentials [LKK24]. Traditional vehicular safety assessments often rely on controlled collision experiments and behavioral testing within predictable circumstances. Nevertheless, Autonomous Driving Systems (ADSs), with their advanced operational complexities and need for environmental versatility, necessitate the conception of multifarious testing situations to encapsulate the breadth of challenges inherently [Gao+22]. Moreover, the multiplicity of real-world traffic scenarios implies that physical testing alone cannot encompass all conceivable risk exposures [HSE22]. As a result, an efficacious approach entails the application of computational simulation technologies to fabricate a spectrum of complex virtual scenarios [Cha+19][Wan+22a]. These simulated environments serve as arenas for rigorously scrutinizing and refining the efficacy of the ADSs.

However, the current methodologies for scripting AD test scenarios, utilizing description languages such as Domain Specific Language (DSL), Unified Modeling Language (UML), and Systems Modeling Language (SysML), are predominantly manual and necessitate ongoing maintenance [Mey+21]. This conventional approach exhibits significant deficiencies and limitations. To begin with, the manual composition of DSLs for intricate, multi-actor scenarios is an arduous and resource-intensive endeavor, with developers dedicating substantial effort to ensure scenario accuracy, coherence, and comprehensiveness. Furthermore, DSLs that are manually crafted are susceptible to errors and oversights, potentially resulting in inadequate test coverage or a dilution in testing rigour. For instance, scripting a rudimentary car-following scenario can encompass in excess of 200 lines of DSL code [Lou+22], demanding considerable time for comprehension and upkeep. It is particularly daunting for novices, posing a formidable learning curve.

An additional pressing concern is the swift progression of AD technology, which concurrently escalates the necessity for an evolving array of test scenarios. The task of manually sustaining and modernizing DSLs to parallel these advancements is formidable and fraught with challenges, often culminating in the obsolescence and functional deficiencies of DSLs. Moreover, the reliance on individual expertise and domain-specific insight during the DSL authoring process results in considerable variability among scenarios crafted by different developers, undermining standardization and reusability [Li+22]. Consequently, the industry is in exigent need of an automated DSL generation mechanism—one that exhibits an understanding of developers' intentions while delivering convenience, automation, standardization, and reusability.

In response to these challenges, scholarly endeavors have initiated the exploration of methodologies for the automatized generation of DSLs tailored to AD testing scenarios. LawBreaker [Sun+22] innovatively converts traffic regulations into driver-centric Signal Temporal Logic specifications and employs a fuzzy testing engine to uncover diverse modalities of rule violation by optimizing specification coverage. However, the instigation of initial scenarios persists as a manual process. Contrarily, the avant-garde world model, DriveDreamer-2 [Zha+24] circumvents the need for conventional DSL representations and leverages LLMs to decipher user requirements, generating detailed multi-perspective video scenarios via diffusion models. Notwithstanding, this approach does not facilitate a closed-loop interaction between the tested AD vehicle and the traffic environment, and its application is confined to the utilization in end-to-end ADS, with planning as the central objective [Hu+23].

To reconcile the automation imperatives with empirical research advancements, we introduce an innovative framework for automated simulation testing scenario generation, designated as Text2Scenario (T2S). Initially, we disentangle complex traffic scenarios into fundamental components and establish a hierarchical scenario repository, ensuring comprehensive element coverage for the construction of the material library. Subsequent to that, we meticulously engineered a prompt-driven workflow and devised a testing text parser predicated on an LLM, enhancing its proficiency in apprehending and deciphering natural language descriptions of test scenarios. In particular, to navigate through the intricacies and ambiguities inherent in natural language, the LLM adopts an approach predicated on multi-stage in-context few-shot

learning. This methodology enables the direct selection of congruent components from the scenario material repository that resonate with the textual descriptions, thereby facilitating the seamless generation of accurate scenario representations. Following this, we establish a tailored DSL corpus—comprising controllable scenario parameters and events—to serve as a viable material repository for the targeted scenario description document. Grounded in the scenario representation, we leverage static element matching and dynamic element concatenation techniques set within a priority-based assembling architecture. This approach meticulously modulates the scenario parameter values within the seed DSL document and introduces segments for event management, culminating in a precise and standard scenario description file. To conclude the process, we customize evaluation indicators specific to the ADS under test. These metrics enable the real-time evaluation of the ADS in the simulation platform, and ultimately yielding comprehensive testing reports.

Our contribution lies in the three folds:

- We introduce a novel framework, T2S, for automated testing scenario generation, which is segmented into five distinctive stages and predicated on textual descriptions to enable virtual simulation testing of ADS;
- To the best of our knowledge, our work represents the inaugural effort to take advantage of the capabilities of LLM for the parsing of intricate natural language scenario descriptions within the realm of DSL and the subsequent generation of standardized, manipulable scenario representations;
- Harnessing a spectrum of test scenario descriptions, we conducted an extensive series of simulation experiments on the Carla platform, which led to the identification of 533 driving safety violations in ADS within 378 generated simulation scenarios. The thorough analysis of these test reports is instrumental in revealing technical vulnerabilities within ADS.

The structure of this paper is organized as follows. In Section 2, we examine the current industry demands and envision potential innovations through the application of LLM. Section 3 introduces our proposed framework for generating test scenarios. The research problem and architecture of the experimental setup come under scrutiny in Section 4. We share the results of our experimental investigations in Section 5. A critical review of the pertinent literature is conducted in Section 6, followed by a discussion on threats to validity of the work in Section 7. In the concluding Section 8, we synthesize the findings of the study and propose directions for future exploration.

2 Motivation

The current landscape of AD scenario testing is encumbered by labor-intensive and time-consuming manual processes that involve translating functional scenarios detailed in natural language into explicit specific test parameters [MBM18][Zha+22], encoded into error-free DSL files [ASA21a]. There is an imperative requirement for an automated and controllable simulation scenario generation methodology that enhances scalability, open-endedness, standardization, and compatibility.

On the contrary, LLMs with powerful natural language generation capabilities [Xu+22], offer an automated solution, capable of rapidly generating complex and varied test scenario representations based on succinct text prompts. When juxtaposed with traditional manual and simulation methods, LLMs wield significant advantages such as:

- *Efficiency:* the ability to swiftly produce a substantial volume of test scenarios;
- *Diversity:* the generation of a broad spectrum of scenarios ensuring extensive coverage;
- *Interpretability:* formulation in natural language that is straightforward to comprehend and modify;
- *Cost-effectiveness:* obviating the need for extensive manual efforts.

To encapsulate, LLMs hold considerable promise in supplanting the manual interpretation of human natural language, engendering standardized and adjustable scenario representations, thereby substantially expediting the testing processes for ADS.

3 Methodology

This section delineates the proposed framework, T2S, for testing scenario generation. Commencing with the insertion of the test language text into the LLM-empowered parser, it comprehends the user's specifications and outputs the scenario representation leveraging the hierarchical scenario repository. Following this, the DSL-based scenario generator is employed to fabricate standardized scenario files. These files are subsequently fed into the simulation environment, whereupon critical evaluative metrics are real-time monitored.

3.1 Overview

Fig.1 elucidates the T2S testing scenario generation framework advanced in this research. The T2S system transmutes testing descriptive text into simulator-ready DSL scripts across five distinct stages, anchored by the ASAM OpenScenario [ASA21a] textbook approach. The first stage necessitates listing the sextet of pivotal traffic scenario elements, in accordance with the Safety of The Intended Functionality (SOTIF)[Wu+21] tenets, which will be exhaustively expounded upon in Section 3.2. Notably, vehicle-to-everything communication level is exempted from this discourse, as it falls outside the scope of this paper.

The secondary stage witnesses the LLM empower the testing text components, utilizing a knowledge base of scenario descriptions to decode the testing text posited by users, identifying the relevant hierarchical pre-configured elements. The following stage proceeds to match the scenario primitive, employing a priority-based assembling methodology to concatenate scenario descriptors into a coherent DSL script.

Ultimately, this scenario descriptor file is integrated into the simulation platform, which interprets and executes the scenario. This integration supports the validation and verification of various AD stacks during the execution phase to yield testing report results.

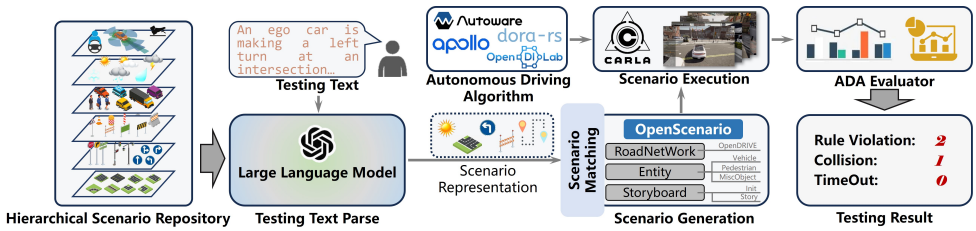

Fig. 1 Overview of the T2S testing scenario generation framework.

3.2 Hierarchical Scenario Repository

Given the presence of numerous intricate components within traffic scenarios, such as ambient weather, road topologies, and unforeseen road debris, it becomes imperative to systematically categorize these elements to concisely define and confine them to a controllable domain [Wan+22b]. Nonetheless, the quantification and description of many such elements present significant challenges [WPC20], particularly when it comes to the subjectivity of participants' yielding behaviors or the unpredictable driving styles of individual drivers. Conventional DSLs [Fre+19][QBC19] necessitate precise and stringent parameter configurations to construct fully functional logical or concrete scenarios. Such rigidity in data specification inadvertently restricts the scenario's malleability, consequentially limiting the exploration of potential flaws in the system under test (SUT). To surmount this limitation, we advocate for a more versatile scenario description language designed to enhance adaptability, empower exploratory changes, and potentially transcend the pre-established confines of stringent DSL scripts at the nascent stages.

The SOTIF-based scenario conceptualizes a hierarchical scenario framework in a systematized and canonical format, delineating interconnections between diverse components, thereby engendering logical affiliations and a tiered structure. Integration of static and dynamic elements is imperative, as the latter are contingent upon the former for contextual grounding. The communication status is intentionally discounted, given our singular emphasis on the self-driving entity. The 6-tiered edifice of hierarchical scenario depiction is assembled layer upon layer, ascending from foundational components to nuanced particulars, as depicted in Fig.2. Leaning on the pre-crash documentation by the National Highway Traffic Safety Administration (NHTSA) [NSY+07], OpenX-Ontology [ASA21b], and the Traffic Safety Handbook [Tex22], Tab.1 enumerates potential elements for every component—road topology, for instance, might encompass intersections, roundabouts, and T-junctions. These elements harbor the capacity to encapsulate extensive semantic connotations, embracing the overwhelming majority of typical scenario constituents encountered in real world. Their ordered combination yields a comprehensive emulation of virtually any envisaged target scenario.

- *Road Topology:* Establishes the foundation upon which all scenario elements rest;
- *Transportation Facilities:* Erected atop road topologies, these structures facilitate transport;

Table 1 Examples of optional elements for hierarchical scenario repository.

Component	Element
Road Topology	Topology: intersection, roundabout, T-junction, etc. Lanes: single lane, two-lanes, three-lanes, etc.
Transportation Facilities	Road Markers: solid line, double solid line, broken line, etc. Traffic Signs: traffic light, stop sign, speed limit sign, etc.
Temporary Changes	Type: cone barrel, warning sign, warning bucket, etc. Position: front, left, right, etc.
Traffic Participants	Type: car, truck, van, etc. Position: front, left, right, etc. Oracle: longitudinal (yield, accelerate, decelerate), lateral (keep lane, change lane), global behavior (go forward, turn)
Climate	Weather: type (sunny, rainy, snowy) and density (strong, medium, weak) Time: daytime, nighttime, morning, etc.
Ego Vehicle	Type: car, truck, van, etc. Position: roadside, right lane, left lane, etc. Global Behavior: go forward, turn left, turn right, etc.

Fig. 2 Hierarchical scenario description. The communication status level is ignored from consideration as our focus is exclusively on the self-driving AD stack.

- *Temporary Changes:* Temporary modifications to road structures and transportation facilities;
- *Traffic Participants:* Encompasses all entities within the background traffic flow, characterized by their static attributes and dynamic behaviors;
- *Climate:* Represents the array of environmental factors within the scenario that influence the preceding components as well as the functionality of vehicle entities;
- *Ego Vehicle:* Leverages the constructed external scenario's first five tiers to influence information and actions occurring within the vehicle's interior;

3.3 Testing Text Parsing

Building on unparalleled advancements in context comprehension, textual synthesis, and image-text correspondence, LLMs like GPT-4 [Ope22] have ascended at a trajectory potentially rivaling or surpassing human expertise. LLMs are on the cusp of supplanting human roles in knowledge interpretation, contextual alignment, and generation of experiential narratives [Las+23] within the domain of standardized process creation. Consequently, within the T2S framework, we investigate the deployment of LLMs as testing text parser, transfiguring user-generated linguistic descriptions into definitive scenario representations.

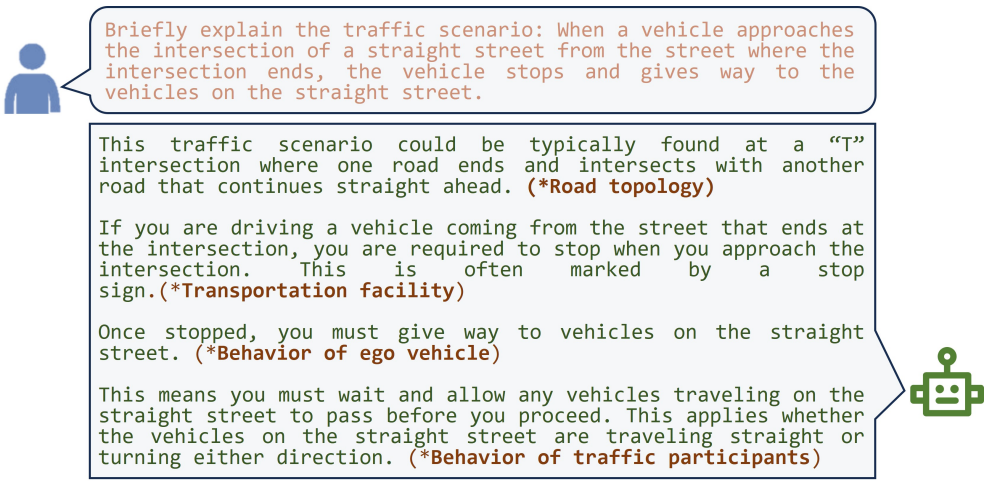

Fig. 3 A case of using prompt for interaction with GPT-4.

(1) *Prompt engineering:* Following thorough training, LLMs have amassed a knowledge repository approximating human expertise. A minimal prompt can actuate these LLMs to assimilate new information and exhibit user-anticipated outputs without necessitating alterations to their intrinsic weighting. This necessitates meticulous prompt engineering to remold the functionality of LLMs. In one case study, we illustrate the interaction with GPT-4 by inputting a scenario description prompt. Referenced in Fig.3, the case study demonstrates that GPT-4 is capable of apprehending a succinct scenario narrative and identifying critical elements within. GPT-4's ability to infer underlying road topologies, such as a "T" intersection, and discern the dynamics of transportation facilities, ego vehicle behavior, and traffic participant activities from a concise description is based on prior-knowledge.

Yet, while LLMs can distill key information from natural language, their linguistic outputs are not currently suitable for direct use in constructing testing scenarios. The incomplete coverage of components in Tab.1 and the introduction of uncertain elements may prevent executable scenarios or produce imprecise guidance, leading to distortions. Furthermore, the conversion of unbounded unstructured output into the hard code format is challenging.

To address these issues, we refined a prompt pipeline tailored for testing text parsers, depicted in Fig.4. This pipeline guides LLMs in generating structured outputs that align with DSL requirements. The Prompt Engineering process encompasses five stages: Role Setting, Few-Shot, Chain-of-Thought, Syntax Alignment Checking, and Self-Consistency. Each stage employs carefully engineered prompts to elicit stable output from the LLM. Within the first stage, Role Setting acts as the LLM's prefix prompt, positioning the LLM as an AD testing expert to refine knowledge retrieval and enhance the quality of responses, an example of which is shown in Fig.5.

(2) *Few-Shot (FS):* To construct the foundational prompt that harnesses in-context few-shot learning paradigm [Liu+24], we initiate by embedding the scenario repository

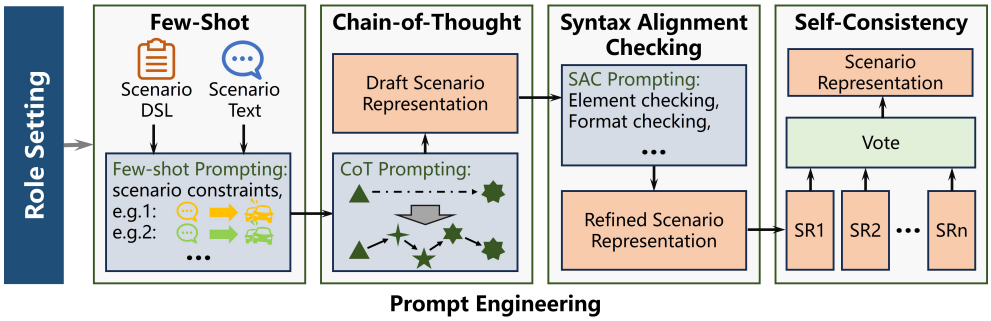

Fig. 4 Pipeline of the prompt engineering embedded in testing text parsing.

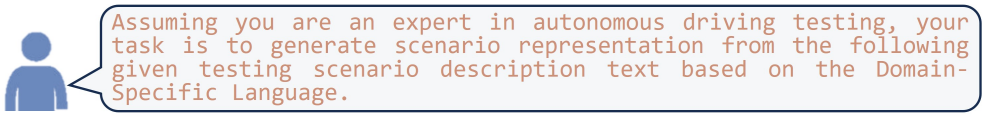

Fig. 5 Prompt of role setting of LLM.

as the selectable knowledge base for scenario elements within the LLM. This approach enables the LLM to comprehend and reproduce structured scenario representations. We facilitate this learning process by supplying a series of input-output case pairs, each consisting of scenario narrative text and its corresponding structured representation. These example pairs act as templates, providing the model with paradigms of standardized structured outputs. Ultimately, we introduce the scenario narrative that necessitates translation into structured format. This text serves as the basic prompt trigger, which, when combined with the aforementioned input-output examples, is furnished to the LLM. This process is graphically encapsulated in Fig.6, illustrating how the LLM synthesizes the provided information to generate structured scenario representations.

(3) *Chain-of-Thought (CoT):* Robust logical reasoning stands as a pillar of the "Intelligence Emergence" exhibited by LLMs [Wei+22], with the crux of reasoning hinging upon the enhancement of their thought processes. Reliance solely on an LLM's inherent knowledge repository may prove inadequate for resolving emergent and unique challenges. Often, the diminished certainty in an LLM's outputs can be attributed to intricate cognitive operations challenging to navigate for the sophisticated tasks. For instance, when tasked with mapping the scenario "unprotected left turn for traffic vehicle" into a corresponding scenario representation, LLMs might not autonomously infer the human-favored action "yield". Instead, it may default to a "decelerate" action, erroneously conflating lower velocities as yielding.

Nonetheless, the capability of LLMs to assimilate new knowledge through systematic prompt induction—without alterations to their internal weights—is significant. By instilling fundamental thinking steps akin to those of a novice, LLMs can be trained

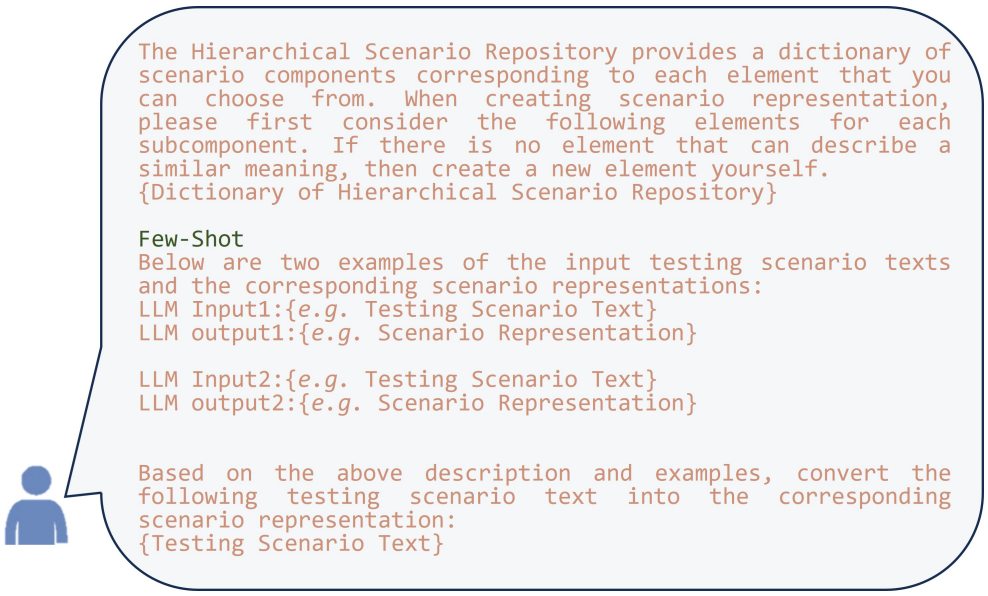

Fig. 6 Prompt of few-shot of LLM.

to produce outputs that more closely align with human reasoning, thus unraveling complex tasks beyond the scope of mere few-shot prompt training.

As illustrated in Fig.7, we dissect the given scenario text "Unprotected left turn for traffic vehicle," extracting pertinent terms and individually scrutinizing them to aid the LLM in a more deliberate contemplation of the natural language stimuli presented. This careful analysis lays the groundwork for the LLM to formulate an appropriate scenario representation, hewing closely to the specified testing scenario via tailored instructional prompts.

(4) *Syntax Alignment Checking (SAC):* Upon the generation of a scenario representation by the LLM, it is imperative to conduct both knowledge validation and syntax harmonization on the output. Knowledge validation involves cross-examining the LLM's conceptual grasp against the original scenario text to ensure congruity. Concurrently, syntax harmonization is employed to refine the output into structured data. To illustrate, an LLM may render the understanding of "straight forward" in a given scenario, while the equivalent term within the scenario repository might be catalogued as "go forward". This minor discrepancy, as minute as a singular word, could thwart hard-coded matching algorithms from accurately identifying the corresponding DSL corpus.

While precision in matching terms is crucial, it is also essential to preserve the intrinsic adaptability of the LLM. The model's unique cognitive potential may extend beyond human contemplation, permitting insight into innovative traffic elements previously unconsidered. In instances where the LLM upholds its original output due to the absence of comparable semantics within the repository, such uniqueness should

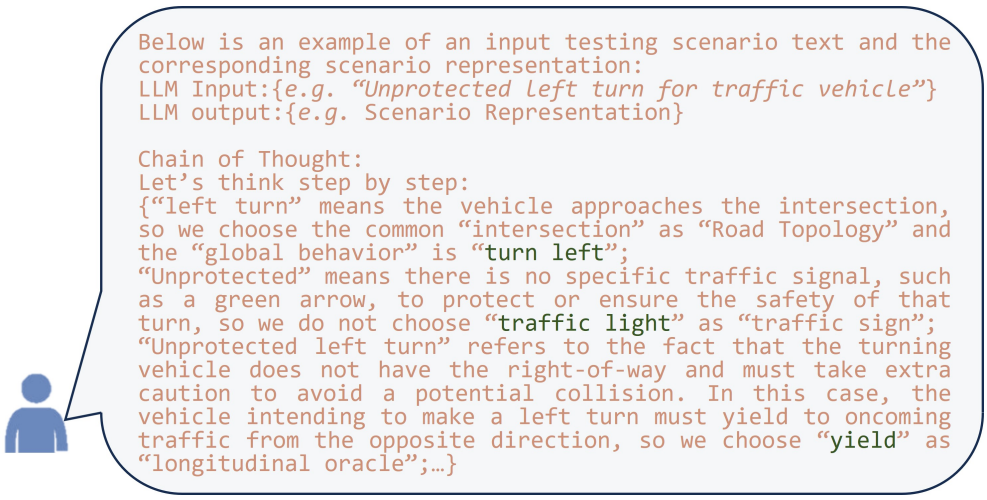

Fig. 7 Prompt of chain-of-thought of LLM.

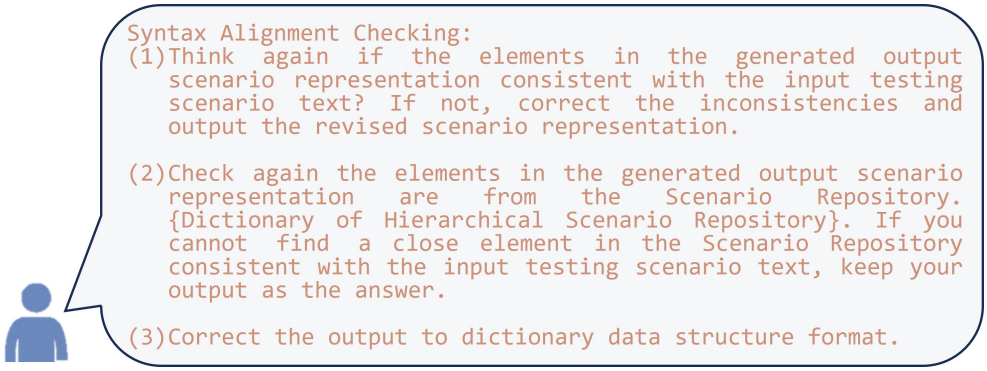

Fig. 8 Prompt of syntax alignment checking of LLM.

be evaluated, not immediately overridden. Furthermore, the structuring of LLM output into a definitive data format is vital, sidestepping the need for laborious manual alignment during intermediary stages. It ensures that the outputs are not only accurate and reflective of the LLM's intelligence but also readily assimilable within the targeted simulation frameworks.

(5) *Self-Consistency (SC):* Given the inherently stochastic nature of deep learning models, the outputs produced by an LLM may exhibit a degree of randomness, leading to potential misalignments with human performance expectations. This entails a balance between response diversity and stability, modulated by hyperparameters such as temperature settings and top-k (nucleus sampling) criteria. To address this variability, a self-consistency strategy is employed. This approach involves the generation of several independent reasoning chains from the LLM. Diverse responses are extracted

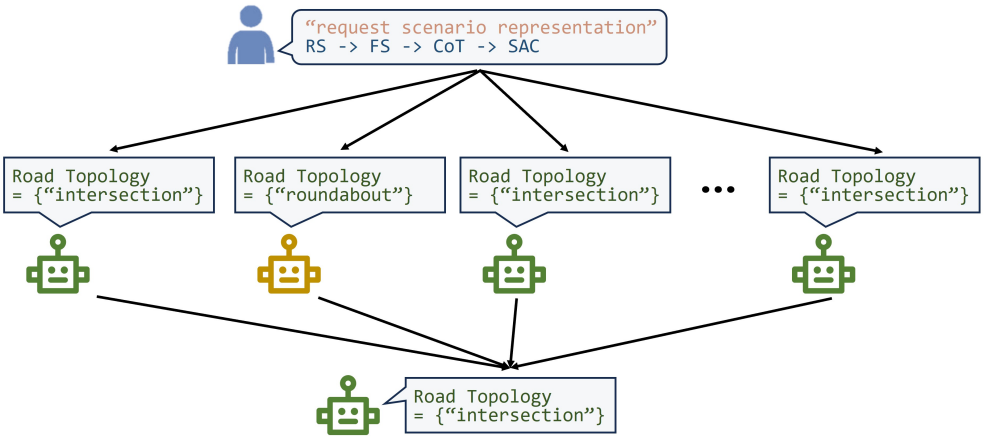

Fig. 9 A case of self-consistency of LLM.

from each chain and the predominant result is determined through a consensus mechanism (or rather, majority vote [Wan+22c]), drawing parallels with the deep ensemble technique [Rah+21] in deep learning to mitigate the unpredictability inherent in a singular model output. The concrete steps consist of drawing distinct inference trajectories from the LLM, followed by a marginalization process that consolidates responses to yield a final answer. As depicted in Fig.9, the graphical illustration represents the self-consistency approach, where multiple divergent reasoning pathways converge to a dominant output labeled as the "intersection". By canvassing a spectrum of potential solutions, the LLM increases the likelihood of generating accurate or significant responses to eliminate potential biases, particularly when confronted with complex tasks that challenge CoT techniques.

3.4 Domain-Specific Language-based Scenario Generation

Acquisition of a DSL-encoded scenario description file amenable to execution by AD simulation software necessitates a systematic translation of the LLM-output scenario representation. This paper opts for the *.xosc* file format, adhering to the ASAM OpenScenario standard. The task involves meticulous link-tuning, effectively correlating the key-value pairs present within the scenario representation with their respective semantic segments as defined by OpenScenario's specification.

This study introduces a priority-based assembling architecture (**Algorithm 1**) tailored for static element matching and dynamic element concatenation between scenario representations and DSL corpus fragments. Commencing with the initialization of a DSL material library—the product of meticulous manual curation—this repository serves as a resource pool for populating target DSL-based files. It comprises essential event fragments such as acceleration, deceleration, lane-changing, and synchronization maneuvers amongst others [ASA21a]. The scenario representation, standardized into a dictionary-format result (*e.g.*, *.json*), is derived from the LLM-powered parser as

expounded upon in Section 3.3. Action chain is an isolated sequence of maneuvers performed by traffic entities. The objective is to delineate each standard action originating from a rudimentary scenario representation of background traffic, thereby crafting an interpretable, finely-tuned DSL control sequence. Take, for instance, the lateral movement component within a traffic participant's scenario representation labeled as "overtaking". Such a label, while conceptually concise, encompasses a suite of discrete standard DSL actions—essentially, "change lanes - accelerate - change lanes - continue at speed"—which collectively constitute the complex maneuver traditionally recognized as "overtaking".

Algorithm 1 Priority Ranking-based DSL Padding

Require: DSL material library $\mathcal{L}$, target DSL-based file $\mathcal{T}$, LLM scenario representation $\mathcal{R}$, action chain of traffic participants $\mathcal{A}$

▷ DSL Material Library Filling

1: $\mathcal{L} \leftarrow$ **Simulator Element PriorKnowledge**
2: $\mathcal{L} \leftarrow$ **Route Random Search**

▷ Static Element Matching

3: $\mathcal{T}$.**Climate**$' \xleftarrow{\mathcal{R}.Climate} \mathcal{L}$
4: $\mathcal{T}$.**Topology** $\xleftarrow{\mathcal{R}.Topology} \mathcal{L}$
5: $\mathcal{T}$.**TransportationFacilities** $\xleftarrow{\mathcal{R}.TransportationFacilities} \mathcal{L} \sim \mathcal{T}$.**Topology**
6: $\mathcal{T}$.**TemporaryChanges** $\xleftarrow{\mathcal{R}.TemporaryChanges} \mathcal{L} \sim \mathcal{T}$.**Topology**
7: $\mathcal{T}$.**EgoVehicle** $\xleftarrow{\mathcal{R}.EgoVehicle} \mathcal{L} \sim \mathcal{T}$.**Topology** & $\mathcal{T}$.**TemporaryChanges**
8: $\mathcal{T}$.**TrafficParticipants**.(**type**&**position_relation**) $\xleftarrow{\mathcal{R}.TrafficParticipants} \mathcal{L} \sim \mathcal{T}$.**Topology** & $\mathcal{T}$.**EgoVehicle**

▷ Dynamic Element Concatenating

9: $\mathbb{A} \leftarrow$ LLM powered action-chain decomposition
10: **for** a in $\mathbb{A}$ **do**
11: $\quad \mathcal{A} \xleftarrow{\mathcal{R}.TrafficParticipants.oracle} a$
12: **end for**
13: $\mathcal{A} \xleftarrow{\mathcal{R}.TrafficParticipants.global\ behavior} \mathcal{A}.append(\mathcal{L}.autopilot())$
14: $\mathcal{T} \leftarrow \mathcal{T} + \mathcal{L}(\mathcal{A})$
15: **return** $\mathcal{T}$

Subsequent to data acquisition from the scenario simulator, the repository necessitates enrichment (refer to lines 1-2). This encapsulates the assimilation of prior knowledge of simulator components, for instance, APIs of weather configuration and the choice of scenery maps. The goal is to supply route-based candidates for evaluation involving the ego vehicles and background traffic. This is actualized by generating, in a stochastic manner, vehicular pathways—from start to target points—within the range of practicable waypoints provided by the simulator, and by annotating these points precisely. As illustrated in Fig.10, envisage a scenario set within a two-laned junction, it is requisite to assign the randomly generated origin and termination coordinates with pertinent labels denoting the intended interaction patterns vis-à-vis the

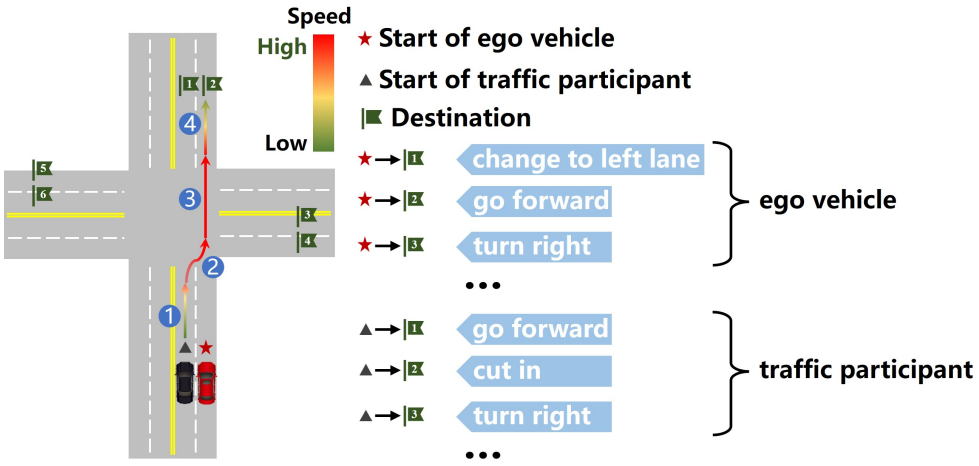

Fig. 10 Schematic diagram and labels for driving route search at intersection.

ego vehicle. Each pair of route extremities, along with their associative labels, are to be integrated into the repository, setting the stage for further synergistic correlation with the scenario representation's key-value pairs.

Subsequently, the DSL-based file designated for scenography was methodically populated with corresponding semantic units to articulate the scenario (refer to lines 3-14). The process commenced by sorting the static elements based on priorities within the hierarchical scenario repository, thereby ensuring congruity in element matching and averting redundant designations (as outlined in lines 3-8). For example, when "intersection" is delineated within ⟨road topology⟩, the ⟨road marker⟩ ought not to be signified as "broken lane," given that vehicular lane transitions at intersections are typically regulated to "solid lane" in compliance with traffic laws. Any contradictions arising in the scenario paperwork would suggest either a faltering in LLM's parsing or an erratum within the scenario's linguistic text, meriting manual scrutiny for clarification. Proceeding from lines 3 to 8, the protocol harnesses the precedence of the static hierarchy; materials are selectively retrieved from the library and embedded into the DSL-based file as delineated by the scenario narrative. The higher the hierarchical standing, the earlier its match is sought. Climate and Topology are bestowed the pinnacle of precedence. Illustrated by the expression $\mathcal{T}.\textbf{TransportationFacilities} \xleftarrow{\mathcal{R}.TransportationFacilities} \mathcal{L} \sim \mathcal{T}.\textbf{Topology}$, it is interpreted as deriving the pertinent fragment from $\mathcal{L}$ to the key within ⟨road topology⟩ and ⟨traffic sign⟩ located in $\mathcal{R}$. The parameters of these segments are then calibrated or a match is sought with a corresponding fragment, predicated on the value associated with the key. And the fragment elected based on $\mathcal{L}$ is subsequently transcribed to the correlating site within $\mathcal{T}$. The symbol "$\sim$" signifies adherence to a higher hierarchical parameter within the $\mathcal{T}$'s Topology.

For the residuum encompassing dynamic elements such as **TrafficParticipants.(oracle & global behavior)**, an approach deploying action dissection and concatenation is utilized, as detailed from line 9 to 14. Fig.10 delineates an instance where the "cut in" maneuver executed by black traffic vehicles in relation to the ego vehicle can be resolved into a sequence of interconnected actions, typified by "acceleration, right-lane change, cruise, deceleration". This sequenced behavior is further deconstructed into a continuum of action chains through the use of LLM, aligning each with their respective, intricate event fragments residing in the library. Post-completion of the action chain, in scenarios where the vehicle is yet to arrive at its target terminus, the automated navigation algorithm embedded within the action chain — as described in line 13 — dictates an appendage from $\mathcal{L}$, propelling the vehicle to fulfill the comprehensive task. To encapsulate the process, the specific $\mathcal{L}(\mathcal{A})$ fragment, representing the action chain, is integrated into the designated locus within $\mathcal{T}$.

3.5 Evaluation Metrics for SUT

Appropriate evaluation metrics can be incorporated into the target DSL-based file to activate the simulator's monitoring system, which persistently scrutinizes the SUT behavior on a frame-by-frame basis throughout the entire execution of the scenario. These evaluative indicators encompass:

- *Rule violation.* This metric ascertains if the SUT contravenes traffic regulations, which entails assessments such as running stop test, running red light test, wrong lane test, and lane keeping test. For instance, a lane invasion sensor could be configured within the Carla simulation for vigilant surveillance.
- *Collision.* This metric determines if the SUT has experienced any collisions with other vehicles or pedestrian objects in its vicinity. Implementing a collision detection sensor within Carla would facilitate perception and aid in recording such events.
- *Time out.* This metric gauges the SUT's ability to navigate to the proximity of the designated endpoint within the stipulated maximal time limit. This timeframe is adaptable and is predicated upon the length of the ADS route as well as the velocity constraint factors inherent within the scenario.

4 Simulation Experiment

In this section, we validate the benefits of the T2S framework using comprehensive textual inputs and structured simulation experiments. The implementation utilizes the ASAM OpenScenario standard format [ASA21b]. The simulations are conducted using the Carla [Dos+17] simulator, and we employ a customized DSL file parser within our refined Carla scenario execution environment.

4.1 Research Questions

To ascertain the efficacy of the T2S framework, we designed three research questions (RQs) targeting key aspects of scenario generation performance. RQ1 examines the extent to which the LLM-powered testing text parsers align with the granularity of

different scenario groundtruth. RQ2 investigates the fidelity with which T2S reproduces scenarios from testing description texts. RQ3 explores the effectiveness of the T2S evaluation mechanism in assessing the driving conduct of diverse SUTs.

- **RQ1: How effectively does T2S's testing text parser perform at capturing the nuances across various scenario complexity levels?**
- **RQ2: Is T2S capable of accurately translating comprehensive testing description texts into executable scenarios?**
- **RQ3: Does T2S facilitate precise evaluation of the driving behaviors for various of SUTs?**

4.2 Benchmark

4.2.1 Testing Text Benchmark

To ensure the scientific rigor of the testing text inputs, we have selected three exemplary scenario description languages that are illustrative of the industry standards: (1) The pre-collision scenario descriptions outlined by the National Highway Traffic Safety Administration (NHTSA) [NSY+07]; (2) The management regulations released in the Active Safety Advanced Driver-Assistance Systems (ADAS) Test Protocol, 2024 Edition, by the China New Car Assessment Program (C-NCAP) [C-N24] and (3) customized natural language texts for testing scenarios, meticulously crafted by testing experts in the field.

It is important to emphasize that not every test scenario provided by the NHTSA and C-NCAP is applicable to our research. Scenarios that did not pertain to the evaluation of ADS performance or that were incompatible with simulation capabilities were omitted from the study. After a careful selection process, a total of 92 test texts were retained for our analysis (NHTSA: 23; C-NCAP: 17; Customized: 52).

4.2.2 Test Scenario Benchmark

Owing to the versatile mapping of a single testing text to various locales within the diverse map topographies in Carla, a solitary test narrative is capable of spawning multiple analogous scenarios. Utilizing the 92 curated texts, we actualized the generation of 368 test scenarios, encompassing all six intrinsic road environments within Carla, which span urban, suburban, and rural settings.

4.2.3 Systems under Test

We selected four distinct types of AD stacks as SUTs, that is, Autoware [Kat+18], Apollo [Apo21], Interfuser [Sha+23], Dora-RS[1]. Autoware and Apollo are recognized as the most sophisticated and widely-adopted application-level AD systems within the industry. Interfuser, developed by OpenDILab, is an advanced end-to-end ADS based on sensor fusion[2]. Doara-RS presents a revolutionary robotic framework, infusing modernity into robotic applications; it is a prototype that hinges on a purely visual

[1] https://github.com/dora-rs/dora.
[2] https://leaderboard.Carla.org/leaderboard/. Interfuser ranks first in the Carla Leaderboard.

modality for its AD capability and boasts a more flexible communication framework than ROS2[3].

4.2.4 LLMs used in Text Parse

Heterogeneous LLMs: For the task of test text parsing, we employed three principal LLMs, including Yi-34B-Chat [You+24], OpenAI GPT-3.5 [Ope22], GPT-4 [Ope22]. In alignment with the protocol delineated in section 3.3 (refer to Fig.4), the deployment of these LLMs was facilitated through their respective official APIs, ensuring systematic invocation of the LLM agents for parsing responses.

Ablation study: To elucidate the significance of each module in the parser, ablation studies were conducted, systematically omitting individual modules to assess their impact. The complete sequence of operations for the parser: Basic Prompt (BP) → Few-Shot (FS) → Chain-of-Thought (CoT) → Syntax Alignment Checking (SAC) → Self-Consistency (SC), enabled the creation of five distinct parser configurations for the purpose of ablation study. These configurations are denoted as LLM-BP, LLM-BP-FS, LLM-BP-FS-CoT, LLM-BP-FS-CoT-SAC, and LLM-BP-FS-CoT-SAC-SC respectively.

4.3 Experiment Settings

4.3.1 RQ1.The matching degree across various levels of detail with the LLM-powered parser

To appraise the hierarchical parsing capabilities of our testing scenario parser—specifically the degree of concordance between scenario representations and the decomposed elements of the testing text—a comparative analysis between the generated representations and the established groundtruth is imperative. In exploring the generation of virtual testing scenarios, we have examined cutting-edge methodologies recently introduced, such as RMT [Den+21] and TARGET [Den+23], for text-based scenario generation. RMT was deemed incongruent as a baseline methodology since it is limited to the generation of video-based open-loop test scenarios, without the provision for a closed-loop feedback model essential for virtual simulation testing. Conversely, TARGET, with its focus solely on parsing traffic rule inputs, fell short of the requisites for crafting the comprehensive virtual scenario testing scripts central to our evaluation framework.

4.3.2 RQ2. The effectiveness of T2S in generating scenarios - feasibility, accuracy, and efficiency

To gauge T2S's impact on mitigating the manual labor involved in constructing testing scenarios, we enlisted four ADS testing experts to juxtapose human efforts with different LLM-based implementations of T2S, thereby providing a comprehensive assessment of their feasibility, accuracy, and efficiency.

[3]https://github.com/ros2/ros2.

For the evaluation of the feasibility of DSL files, we implemented a monitor to track and log any read or run-time errors that may occur in the generated OpenScenario DSL files.

To appraise the accuracy of the scenarios generated by T2S, we recruited 15 automotive and transportation students to manually assess the congruity between the recorded output videos of the scenarios and their corresponding textual descriptions. They quantified the accuracy of T2S's scenario generation by attributing a match score ranging from 0 to 1 — using a 0.1 discrete interval scale, with higher scores indicating a greater match.

Concurrently, the efficiency of T2S's rapid and precise automated testing scenario generation was evaluated by measuring the time taken for scenario construction.

4.3.3 RQ3. Compatibility of T2S evaluator with heterogeneous SUTs

All 4 varieties of SUTs have been seamlessly integrated to ensure compatibility with the DSL testing files. The configuration for these SUTs is restricted to designating starting and ending points, deliberately precluding human intervention in modulating any intermediate behaviors. Throughout the execution of each scenario, the performance of the SUT is attentively observed, with the objective of recording its evaluation metrics. The data accrued from these observations is then utilized to synthesize comprehensive test reports.

4.4 Evaluation Metrics

4.4.1 RQ1. The matching degree across various levels of detail with the LLM-powered parser

To quantify the LLM-powered parser's proficiency in scenario-level analysis, the correctness of parsing for each constituent sub-element within the scenarios was methodically evaluated. The parser's proficiency at managing and interpreting intricate textual information is reflected in the ratio of accurately parsed elements across the entirety of the test cases. This metric serves as an indicator of the parser's overall analytical capability.

4.4.2 RQ2. The effectiveness of T2S in generating scenarios - feasibility, accuracy, and efficiency

We quantify the capabilities of T2S and human testing experts across three primary dimensions: feasibility, accuracy, and efficiency. Concerning feasibility, the incidence of read errors and run-time errors throughout the scenarios serves as crucial benchmarks for ascertaining the executability level of T2S. The lower, the better. With respect to accuracy, we tasked judges with appraising two critical aspects: 1) Semantic fidelity (0-1)—this assessment gauges the semantic coherence between the input description and the generated scenario; and 2) Driving rationality (0-1)—this evaluation determines the extent to which the traffic participants' behavior in the generated scenario aligns with normal traffic patterns. Regarding efficiency, we elected to utilize the metric of

scenario construction time as a basis for comparison with the proficiency and pace of professional human testing experts in scenario creation.

4.4.3 RQ3. Compatibility of T2S evaluator with heterogeneous SUTs

Throughout the execution phase of the scenario, three critical indicators are selected to assess the competency of the AD stack: rule violation, collision, and timeout. For the timeout criterion, a threshold is established by dividing the total distance of the test route by 10% of the speed limit, which serves as the maximum allowable time for the AD stack to complete the test route.

5 Results

5.1 RQ1. The matching degree across various levels of detail with the LLM-powered parser

Tab.3 delineates the parsing precision of three different LLMs for each element when a comprehensive prompt pipeline is employed, encompassing a total of 92 language descriptions within this study. The accuracy of the scenario representations produced by the LLM-based testing text parser is contrasted with the groundtruth labeled by human experts within the original scenario descriptions. This comparison evaluates their proficiency in comprehending and extracting knowledge from textual language. ChatGPT-4.0 exhibits superior language comprehension abilities relative to Yi-34B-Chat and ChatGPT-3.5, as evidenced by its closer alignment with human-level accuracy in interpreting most scenario elements, attributable to its more advanced pre-trained model. While Yi-34B-Chat outperforms ChatGPT-4.0 in parsing certain elements slightly, it maintains a generally stronger performance than ChatGPT-3.5.

Within the scope of LLM analysis, explicit static expressions such as road topology, climate, and vehicle types are typically comprehended with relative ease. However, LLMs often struggle with the interpretation of complex vehicular behaviors and the nuances of multiple traffic participants' interactions, particularly when dealing with implicit expressions. For instance, the statement "*The ego vehicle encounters a vehicle cutting into its lane from a lane of static traffic*" can be misinterpreted by an LLM, which might fail to infer the implication of numerous stationary vehicles present in that lane, sometimes only generating a single vehicle in response. This issue is frequently linked to the error propagation inherent in the input-output examples used during the few-shot, as we have identified that LLMs tend to yield correct results when provided with similar instances. Moving forward, our research will delve into refining prompt engineering techniques to foster a more encompassing comprehension of traffic components by LLMs.

In a horizontal comparative analysis among the three LLMs, ChatGPT-4.0 holds an average parsing accuracy of 92.3%, which surpasses Yi-34B-Chat and ChatGPT-3.5 by 4.2% and 11.9%, respectively, standing at 88.1% and 80.4%. This advantage is particularly noticeable in understanding dynamic behaviors within oracles, preventing

Table 2 The accuracy of hierarchical scenario element parsing for different LLMs based parsers (378 scenarios). RT-Road Topology, TF-Transportation Facilities, TC-Temporary Changes, TP-Traffic Participants, C-Climate, EV-Ego Vehicle.

Element	Yi-34B-Chat	ChatGPT-3.5	ChatGPT-4.0
RT.⟨topology⟩	**0.96**	0.89	0.95
RT.⟨lanes⟩	0.96	0.85	**0.98**
TF.⟨roadmarker⟩	0.90	0.90	**0.97**
TF.⟨trafficsign⟩	0.99	0.97	**1.00**
TC.⟨type⟩	0.83	0.80	**0.95**
TC.⟨positionrelation⟩	0.81	0.76	**0.92**
TP.⟨type⟩	0.95	0.89	**0.98**
TP.⟨positionrelation⟩	0.83	0.62	**0.83**
TP.⟨longitudinaloracle⟩	0.71	0.59	**0.74**
TP.⟨lateraloracle⟩	0.72	0.53	**0.77**
TP.⟨globalbehavior⟩	**0.86**	0.70	0.83
C.⟨type⟩	0.98	0.91	**0.99**
C.⟨density⟩	0.87	0.68	**0.97**
C.⟨time⟩	0.90	0.86	**0.95**
EV.⟨type⟩	0.99	0.98	**0.99**
EV.⟨position⟩	**0.95**	0.90	0.95
EV.⟨globalbehavior⟩	0.82	0.76	**0.86**
Average	0.88	0.80	**0.92**

significant performance degradation. Looking ahead, enhancing LLM parsing capabilities for certain concealed elements could benefit from a multimodal data analysis approach that integrates text, imagery, and video elements.

5.2 RQ2. The effectiveness of T2S in generating scenarios - feasibility, accuracy, and efficiency

5.2.1 Feasibility

In our assessment, T2S-generated scenarios based on different LLM have a markedly lower success rate when compared to human experts, as elucidated in Fig.11. Human experts achieved near-perfect rates of successful script formation. This is because there is no temporal restriction on scenario construction — experts can repeatedly revise the scripts through an iterative process of reading, comprehending, authoring, debugging, testing, and modifying until they achieve success. In contrast, LLM-based T2S lack the capability to verify the feasibility of their generated scenario representations during the construction process. We have, however, preserved the adaptable operation space of LLM-based systems, aspiring to the principle of meticulous understanding over a rigid, hard-coded approach. Siphoning lessons from the human scenario construction methodology, our future research aims to improve the generation of T2S through closed-loop feedback mechanism.

ChatGPT-4.0 maintains a significant advantage over its counterparts regarding the creation of executable scenarios, securing an 87.31% success rate, which stands considerably higher than the 5.47% and 20.21% achieved by Yi-34B-Chat and ChatGPT-3.5, respectively. Our analysis uncovered that read errors typically arise when LLMs select components that fall outside the established dataset, lacking corresponding DSL

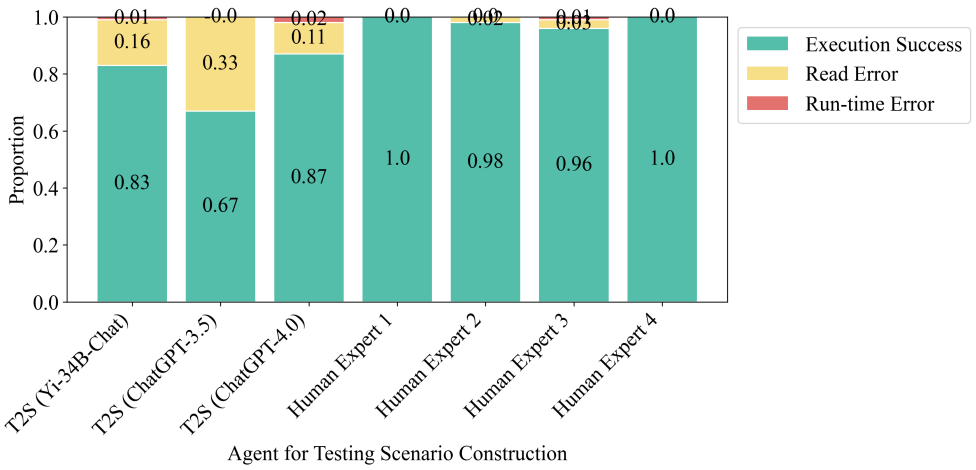

Fig. 11 The statistic proportion of DSL-file made by different LLMs-based T2S and human expert agents.

fragments, which may hinder the generation of accurate DSL files. Runtime errors commonly occur due to abrupt terminations in the runtime, associated with legitimate but imperceptibly contradictory parameter configurations within the DSL text of the test scenario. Human experts, with their extensive experience in fine-tuning scenario parameters, are adept at averting such issues.

5.2.2 Accuracy

For the purpose of evaluating the accuracy of the scenario parser, we utilized the top-performing ChatGPT-4.0. A sampling of 23 scenarios out of the 368 was randomly selected, and human evaluators were invited to rate the scenarios for semantic fidelity and driving rationality. The distribution of these scores is depicted in Fig.12, with further details provided in the supplementary materials[4]. Acknowledging the inherent subjectivity and variability among different human judges — some being more inclined to award higher scores while others favor lower scores for the same item — we employed the Intraclass Correlation Coefficient (ICC) bidirectional random effects model [SF79]. This model serves to assess the consistency or reliability of quantitative data acquired through repeated measures, including multiple evaluators scoring the same variables. The statistical analysis revealed that the ICC value for semantic fidelity stands at 0.92, denoting superb consistency. Meanwhile, the ICC for driving rationality is 0.76, indicative of good consistency. The comparatively lower value for the latter can be ascribed to the diverse driving experiences of the evaluators, which in turn influence their perceptions and interpretations of traffic behavior. For instance, seasoned drivers might anticipate that the conduct of road users should account for interaction with other participants — a factor not encapsulated in the original DSL scenario execution

[4]https://docs.google.com/forms/d/e/1FAIpQLSdUoDLjNLBwgTNYyCRd9KYRjg4RML9LKa-GXPeeKcCOjiSk5g/viewform?usp=sf_link.

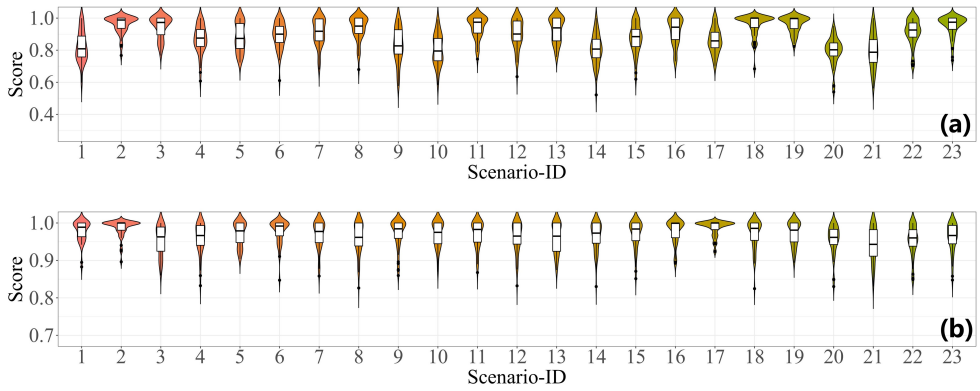

Fig. 12 The scoring distribution of (a) semantic fidelity and (b) driving rationality of 23 generated scenarios from T2S by human judges.

file. Consequently, in certain scenarios, some traffic participants might display driving behaviors that are deemed illogical by these evaluators.

Our analysis extended into the exploration of factors contributing to the lower scores assigned to certain scenarios. It appears that the deficiency in the scenario's graphical representation, particularly the inadequate depiction of water stains on the roadway (as with scenario ID 1 in subset (a)), coupled with the omission of road friction coefficients, resulted in an unrealistic portrayal of the expected loss of control. Similarly, in scenario ID 23 in subset (b), evaluators frequently noted a discrepancy between the exhibited traffic flow behaviors and standardized norms. Notwithstanding these instances, the average scores dispensed by most evaluators for both semantic fidelity and driving rationality were predominantly high, averaging 0.97 and 0.95, respectively. This affirmatively validates the capability of T2S in generating scenarios that are both semantically accurate and adherent to rational driving expectations, underscoring its utility in the intended application domain.

5.2.3 Efficiency

To assess the impact of the T2S framework on lessening the intensity of human labor, we quantified the scenario construction duration for various LLM-based T2S agents and a cohort of four human experts. Demonstrated in Fig.13 as a bar graph with error bars, the results indicate that T2S delivers a marked advantage in terms of efficiency, with construction times consistently falling below 200 seconds. Predominantly, time expenditure is attributed to LLM's cloud-based inference, where the ChatGPT-X series excels in response agility. Comparatively, the mean construction times for ChatGPT-3.5 and ChatGPT-4.0 stand at approximately 15 seconds and 55 seconds, respectively. The Yi-34B-Chat exhibits a longer average construction time, which is around 130 seconds. In stark contrast, the construction time for human experts notably exceeds that of T2S agents, with an average duration surpassing 600 seconds, compounding greater variability in scenario construction times. Although the efficiency

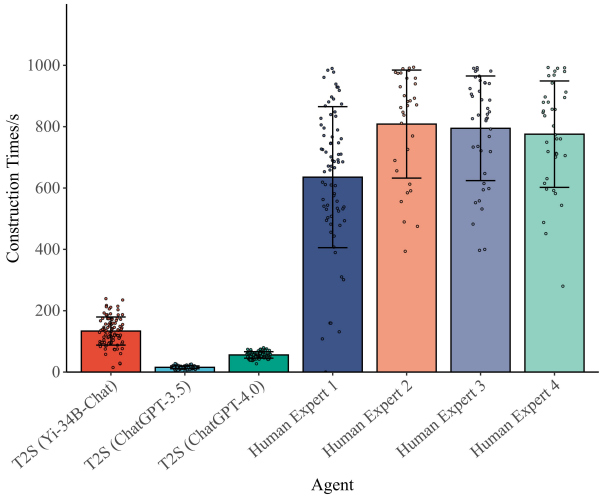

Fig. 13 Bar graph with error bars of construction time of T2S and human expert agents.

of human professionals tends to improve over multiple attempts, achieving the stable processing time exhibited by T2S agents is challenging. Human participants are susceptible to grammatical and detailed errors, necessitating frequent iterative cycles of verification and debugging. In summary, the T2S framework greatly mitigates the time invested in constructing virtual scenarios, thereby diminishing the extent of human labor required.

5.3 RQ3. Compatibility of T2S evaluator with heterogeneous SUTs

Fig.14 delineates the snapshots of testing scenarios for four divergent SUTs-Apollo, Autoware, Interfuser, and Dora RS. The upper portion of the figure emanates from the SUT systems' developer-provided visual interfaces, while the lower segment illustrates the controlled ego vehicle within the Carla. The OpenScenario execution files, paramount for these tests, have been meticulously refined to ensure cross-compatibility with the aforementioned SUT variants. Detailing the SUTs' performance, Apollo experienced a collision, attributable to its deficient recognition capabilities in detecting two-wheeled motorcycle. Autoware, faced with an abruptly reducing target point in Carla's roundabout representation, deviated onto a high curvature trajectory, ultimately transgressing the lane boundary. Interfuser's inadequacy surfaced during a lane-change maneuver involving close-quarters interactions, where it failed to maintain a precise account of its dimensions, leading to minor abrasions. Dora-RS's perceptual mechanisms faltered, lacking the requisite training to detect out-of-distribution, such as road-side vehicles opening their doors, culminating in a failure to recognize the hazard and consequently a severe collision.

Tab.3 meticulously documents the number of safety violations occurrences for each SUT within the extensive ensemble of 378 test scenarios. Apollo and Autoware have displayed commendable rule adherence and reduced collision instances, a testament

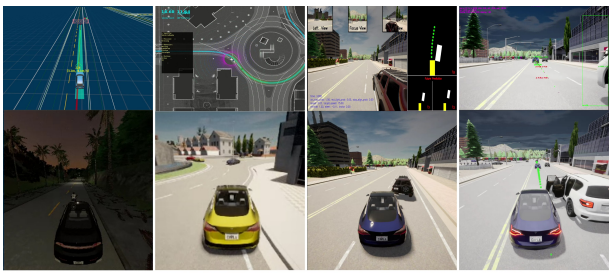

Fig. 14 Snapshots of violation scenarios for four different SUTs. (a) Apollo. Chasing a two wheeled motorcycle with no lights on in the dim dusk; (b) Autoware. Violating the right solid line when entering the roundabout; (c) Interfuser. When changing lanes and overtaking, it collides with the rear end of the same lane; (d) Dora RS. The sudden opening of the door by a parked vehicle on the roadside caused a collision.

Table 3 Number of safety violation events for four different SUTs (378 scenarios, with the maximum number of violations for each type highlighted in bold).

ADS	Rule Violation	Collisions	TimeOut
Apollo	31	23	49
Autoware	37	11	19
Interfuser	**76**	49	**60**
Dora-RS	61	**67**	50

to their elaborate compliance state machines and robust security frameworks that command their behavior. In stark contrast, Interfuser has been identified as the preeminent transgressor with a total of 76 Rule Violations, leading the cohort in this regard. Concurrently, Dora-RS held the undesirable distinction of being the most collision-prone SUT, recording 67 incidents. Analysis attributes this vulnerability to the frequent lapses in its vision-only detection system, which transmits flawed perceptual data, predisposing the system to engage in rear-end collisions or margin-centric incidents evocative of the scenarios delineated in Figure 14 (d). A peculiarity was noted in the comparison between Apollo and Autoware's Timeout occurrences: Apollo presented with a surprisingly elevated rate, disparate from its projected operational capabilities. Investigations attributed this divergence to potential disparities within Guardstrikelab's Apollo-Carla-Bridge software parameters, which may sporadically lose connection, thereby precipitating signal interruption, collisions, or even inducing system inoperability. These findings have been corroborated through discussions within the open-source community forums[5]. Conclusively, under the DSL framework, T2S with Carla as its foundational simulation platform is compatible with supporting multiple SUT tests, thereby ensuring compatibility and support for parallel testing of a myriad of SUTs.

[5] https://github.com/guardstrikelab/Carla_apollo_bridge/issues/86.

Table 4 Ablation studies for the average matching accuracy. (Average matching accuracy=Success rate of execution $*$ Average accuracy of the element parsing.)

LLM	BP	BP-FS	BP-FS-CoT	BP-FS-CoT-SAC	BP-FS-CoT-SAC-SC
Yi-34B-Chat	0.109	0.485	0.644	0.642	0.734
ChatGPT-3.5	0.134	0.416	0.458	0.507	0.542
ChatGPT-4.0	0.548	0.656	0.685	0.765	0.803

5.4 Ablation Studies

To delve deeper into the influence of individual prompt engineering modules within the LLM-based parser, we conducted a series of ablation studies as depicted in Table 4. The analysis elucidates that, horizontally, there's a progressive enhancement in the matching accuracy of parsed outputs with the incremental inclusion of modules, particularly spotlighting the pivotal role of FS, which evidently steers the LLM toward generating outputs adherent to the pre-defined data structure, thereby curtailing compilation errors. For the Yi-34B-Chat, the CoT module markedly amplified its average matching accuracy by 0.159, starkly highlighting its effectiveness. Yet, the SAC module unexpectedly induced a marginal decrement, which could be speculated as stemming from the intrinsic randomness and indeterminacy associated with LLMs [Yin+23]. Nonetheless, this dip was adeptly mitigated by the introduction of the SC module, which, via a voting mechanism, appreciably fortified the stability and congruence of the model's outputs. From a vertical standpoint, when subjected to rigorously crafted prompt instructions, Yi-34B-Chat registered a remarkable ascent in performance, overtaking ChatGPT-3.5 with a range expanding from a -0.025 deficit on the BP to a substantial +0.192 lead when all modules were engaged (BP-FS-CoT-SAC-SC). ChatGPT-4.0 persistently exhibited a superior intellectual performance under analogous prompt conditions, notably achieving a matching accuracy index exceeding 0.4 even in the bare BP condition. Reflecting on the performance trends, ChatGPT-4.0 exhibits a markedly slower rate of degradation in capability across various prompts compared to its counterparts, substantiating its inherently robust general intelligence. In summary, each module distinctly contributes to enhancing LLMs' output proficiency: FS systematically structures the output, CoT augments the interpretive and analytical capabilities for implicit expressions, SAC rigorously scrutinizes output for syntactic and structural integrity, and SC bolsters the overall stability and reliability of LLM-generated outputs.

5.5 Discussions and Limitations

T2S streamlines the construction of scenario DSL files while preserving the fidelity and precision of scenario execution, thereby signifying a tangible reduction in manual labor. Its integration with human expertise is poised to significantly enhance simulation-based testing efficiency in ADS. Nonetheless, T2S is not devoid of challenges that necessitate strategic solutions.

- **Robustness of LLM:** Variations in testing scenario texts, albeit semantically identical, can precipitate disparate scenario representations due to the current fragility of LLM robustness — a longstanding quandary in deep learning domains. Despite the stabilization efforts through employing SAC and SC, the resilience of LLM-based models against inconsistencies demands further refinement. Advancing the reliability of LLMs calls for additional focused training involving meticulous dataset curation and model fine-tuning.
- **Scalability Constraints:** While the current process incentivizes the usage of pre-existing components within the element repository, the hardcoding of DSL files, though beneficial for readability, executability, and controllability, inadvertently restricts the scope for probing uncharted corner cases. Future research endeavors aspire to explore innovative strategies for adaptively generating matching scenarios in the absence of pre-established element library components.
- **Format Limitation:** The present scope of the study confines itself to the generation of OpenScenario standard files. The specificity of the output restricts adaptability to alternative DSL target files, thereby limiting the versatility of T2S. To address this bottleneck, an end-to-end scenario material library generation methodology—which envisages instructing a description file writing model through exposure to a multitude of sceario descriptions—merits exploration. This model could independently author DSL code, shaping a new frontier in AD testing research.

6 Literature Review

6.1 Scenario-based ADS testing

Autonomous vehicles' development and deployment have intensified the need for robust validation and verification methods [Pat+23]. Among the various approaches proposed in the literature [Apa+21][Koo17][Ind+21], one particularly promising avenue is **scenario-based testing**. By formulating scenarios that encompass diverse real-world situations, researchers aim to push the boundaries of testing methodologies.

One of the key aspects of scenario-based testing is the generation of relevant test scenarios. Ghodsi et al. [Gho+21] propose an efficient mechanism to characterize and generate testing scenarios using a state-of-the-art driving simulator. Wang et al. [Wan+22b] utilize in-depth crash data involving powered two-wheelers to generate realistic testing scenarios for autonomous driving systems. Several other works have focused on the generation of challenging scenarios for AV testing. Zhou et al. [Zho+22] employ a genetic algorithm to generate scenarios that increase the probability of collisions or near-collisions, which are deemed as challenging for AVs. Chen et al. [Che+21] introduce an adversarial evaluation framework that generates lane-change scenarios to expose weaknesses in AVs' decision-making policies.

In addition to scenario generation, researchers have explored methods for efficiently searching the vast scenario space to identify critical test cases. Feng et al. [Fen+22] propose a multimodal critical-scenario search method that combines optimization techniques with supervised learning to efficiently find scenarios that expose AV failures. In order to expand the security testing theory in connected environments, Shi et al. [Shi+22] present an integrated traffic and vehicle co-simulation framework that enables

testing of connected and autonomous vehicle technologies, including vehicle-to-vehicle and vehicle-to-infrastructure communication.

Current research predominantly revolves around the identification and analysis of risk scenarios within AD through refinement and application of optimization or learning algorithms, leveraging existing datasets. Yet, this conventional approach necessitates user intervention for the initial establishment of seed scenarios, resulting in a notable absence of fully automated processes capable of producing anticipated scenario configurations derived directly from user-conceived ideas or blueprints.

6.2 LLM-Driven Scenario Generation

The advent of advanced large model technologies in recent years has indeed unlocked new possibilities in automatically generating high-quality datasets, inclusive of input scenarios pertinent to AD [Xi+23]. An emerging research endeavor involves employing LLMs for the simulation and generation of traffic scenarios with increased fidelity. The CTG++ [Zho+23] framework devised by Zhong et al. harnesses the synergy between spatiotemporal transformers and LLMs, empowering users to craft realistic and controllable traffic scenarios via intuitive natural language instructions. Moreover, Li et al. [Li+24] have showcased the potential of LLMs in conjuring traffic scenarios from SUMO configuration files, effectively circumventing the conventional reliance on graphical editor interfaces or the labor-intensive process of manual XML file authorship. Despite these advancements, the extent of controllability within these generated scenarios remains an area deserved for further exploration and validation.

Recent research contributions reveal an intensified interest in harnessing LLMs' capabilities to assimilate traffic regulations and natural language descriptions, transforming them into explicit driving scenarios. Deng et al. [Den+23] and Guezay et al. [GÖK23] pioneered methods wherein LLMs facilitate the automatic distillation of knowledge from traffic laws, spawning driving scenarios congruent with regulatory mandates, which subsequently serve in assessing diverse ADS software stacks. This innovation significantly bolsters automation and broadens the variety of autonomous driving test scenarios [LT23]. In a similar vein, Barone et al. [MLI23] ventured into merging LLMs with automated driving scenario generation, elucidating techniques to construct and refine the driving conduct of AI-powered vehicles predicated on natural language scripts. Extending the application of LLMs beyond the automotive sphere, Cao et al. [CL23] proposed a novel approach to craft robot behavior trees. Their methodology leverages LLMs for designing and autonomously realizing intricate cross-domain tasks through an intelligible behavior tree architecture.

While the domain of scenario generation leveraging LLMs is experiencing rapid growth, it currently faces a shortfall in tailored research aimed at crafting scenarios that align squarely with user expectations and benefit from LLM-assisted unified DSL creation. Distinct from prevailing studies in the arena of scenario generation, our approach capitalizes on the capabilities of LLMs to produce standardized, manageable, and universally applicable scenario description files via linguistic and textual inputs. This methodology ensures compatibility across a broad spectrum of test subjects and environments, marking a significant advancement towards more versatile and user-centric scenario generation frameworks.

7 Threats to Validity

We discussed threats to validity from three aspects: external validity, internal validity, and construct validity, as suggested by the literature [Woh+12].

External Validity: This pertains to the applicability and generalizability of our research methods. By employing a diverse LLM approach, we've automated the generation of test scenarios adhering to the standards of both North American NHTSA and Chinese C-NCAP, alongside expert-customized inputs. These scenarios are characterized by their cross-regional and multi-dimensional attributes and have been implemented within simulation Carla to test ADS that follow different technological paradigms. Thanks to meticulously structured prompt engineering, we achieved an automatic scenario generation accuracy of 80.3%, showcasing the T2S framework's adeptness in comprehending texts, matching elements, and generating scenarios. The final scenario descriptions are formatted in the internationally recognized DSL, OpenScenario, ensuring seamless adaptability to other simulation environments compatible with this standard.

Internal Validity: This concerns the direct correlation between experimental outcomes and our research methodology. Through comparative ablation studies specifically on the prompt module, we observed that the combination of LLM and strategic prompt engineering significantly improves the executable rate and fidelity of autogenerated test scenarios. It reinforces the effectiveness of our approach in generating more reliable and applicable scenarios for ADS testing.

Construct Validity: This relates to the suitability of the ADS and simulators employed in our study. To ensure a broad representation, we selected four ADS technologies known for their distinct operational frameworks. Autoware and Apollo are modular ADS solutions prevalent in industry applications. Interfuser, notable for its high performance in the Carla leaderboard, exemplifies a multimodal end-to-end ADS indicative of potential future trends. Dora-RS, distinct for its reliance on visual processing, surpasses conventional ROS2 communication architectures in optimization. Carla, a widely endorsed open-source simulator, is regularly updated and expanded by its developer community, presenting a rich, high-fidelity simulation environment that vividly replicates real-world transportation contexts.

8 Conclusion and Future Direction

In conclusion, this research elucidates the design and implementation of an advanced automated scenario generation framework, known as Text2Scenario, which is adeptly engineered to process autonomous driving test simulations from natural language descriptions leveraging a sophisticated large language model. The LLM-powered parser stringently selects relevant scenario elements from our extensive hierarchical scenario repository that align with the submitted textual descriptions. The fusion of static and dynamic elements within a structured prioritization matrix subsequently enables the creation of executable DSL files. These files are instrumental in the real-time appraisal of ADS performance. Our rigorous manual evaluation process validates the efficacy of the T2S approach, achieving a commendable success rate of 80.3% in generating accurate simulation scenarios across a diverse range of input texts.

This research stands at the forefront, representing an inaugural exploration into the realm of standardized DSL scenario description file generation via natural language text inputs. As we venture into the future, our focus will pivot towards the formulation of a seamless end-to-end framework for automated scenario file production. This initiative is aimed at dismantling existing barriers associated with testing material databases. Simultaneously, we aspire to incorporate the synthesis of more intricate and high-risk scenarios into our research methodology.

Declarations

- **Acknowledgements** The authors would like to appreciate the financial support of the National Key Research and Development Project of China under Grant 2022YFB4300400.
- **Conflict of interest** On behalf of all the authors, the corresponding author states that there is no conflict of interest.

References

[SF79] Patrick E Shrout and Joseph L Fleiss. "Intraclass correlations: uses in assessing rater reliability." In: *Psychological bulletin* 86.2 (1979), p. 420.

[NSY+07] Wassim G Najm, John D Smith, Mikio Yanagisawa, et al. *Pre-crash Scenario Typology for Crash Avoidance Research*. Tech. rep. United States. National Highway Traffic Safety Administration, 2007.

[Woh+12] Claes Wohlin et al. *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[Dos+17] Alexey Dosovitskiy et al. "CARLA: An open urban driving simulator". In: *Conference on robot learning*. PMLR. 2017, pp. 1–16.

[Koo17] Philip Koopman. "Challenges in autonomous vehicle validation: Keynote presentation abstract". In: *Proceedings of the 1st International Workshop on Safe Control of Connected and Autonomous Vehicles*. 2017, pp. 3–3.

[Kat+18] Shinpei Kato et al. "Autoware on board: Enabling autonomous vehicles with embedded systems". In: *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE. 2018, pp. 287–296.

[MBM18] Till Menzel, Gerrit Bagschik, and Markus Maurer. "Scenarios for development, test and validation of automated vehicles". In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2018, pp. 1821–1827.

[Cha+19] Qianwen Chao et al. "Force-based heterogeneous traffic simulation for autonomous vehicle testing". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 8298–8304.

[Fre+19] Daniel J Fremont et al. "Scenic: a language for scenario specification and scene generation". In: *Proceedings of the 40th ACM SIGPLAN conference on programming language design and implementation*. 2019, pp. 63–78.

[QBC19] Rodrigo Queiroz, Thorsten Berger, and Krzysztof Czarnecki. "GeoScenario: An open DSL for autonomous driving scenario representation". In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2019, pp. 287–294.

[Yan+19] Diange Yang et al. "Driving space for autonomous vehicles". In: *Automotive Innovation* 2 (2019), pp. 241–253.

[WPC20] Mingyun Wen, Jisun Park, and Kyungeun Cho. "A scenario generation pipeline for autonomous vehicle simulators". In: *Human-centric Computing and Information Sciences* 10.1 (2020), p. 24.

[Apa+21] Vimal Rau Aparow et al. "Scenario based simulation testing of autonomous vehicle using Malaysian road". In: *2021 5th International Conference on Vision, Image and Signal Processing (ICVISP)*. IEEE. 2021, pp. 33–38.

[Apo21] ApolloAuto. *Apollo*. 2021. URL: https://bit.ly/2E3vWyo.

[ASA21a] ASAM. *ASAM OpenSCENARIO: User Guide*. 2021. URL: https://www.asam.net/index.php?eID=dumpFile&t=f&f=4092&.

[ASA21b] ASAM. *ASAM OpenXOntology*. 2021. URL: https://www.asam.net/standards/asam-openxontology/.

[Che+21] Baiming Chen et al. "Adversarial evaluation of autonomous vehicles in lane-change scenarios". In: *IEEE Transactions on Intelligent Transportation Systems* 23.8 (2021), pp. 10333–10342.

[Den+21] Yao Deng et al. "RMT: Rule-based metamorphic testing for autonomous driving models". In: *arXiv* (2021), pp. 1–12.

[Gho+21] Zahra Ghodsi et al. "Generating and characterizing scenarios for safety testing of autonomous vehicles". In: *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2021, pp. 157–164.

[Ind+21] Francis Indaheng et al. "A scenario-based platform for testing autonomous vehicle behavior prediction models in simulation". In: *arXiv preprint arXiv:2110.14870* (2021).

[Mey+21] Max-Arno Meyer et al. "Simulator coupled with distributed co-simulation protocol for automated driving tests". In: *Automotive Innovation* 4.4 (2021), pp. 373–389.

[PWL21] Liang Peng, Hong Wang, and Jun Li. "Uncertainty evaluation of object detection algorithms for autonomous vehicles". In: *Automotive Innovation* 4.3 (2021), pp. 241–252.

[Rah+21] Rahul Rahaman et al. "Uncertainty quantification and deep ensembles". In: *Advances in neural information processing systems* 34 (2021), pp. 20063–20075.

[Wu+21] Siyu Wu et al. "A new SOTIF scenario hierarchy and its critical test case generation based on potential risk assessment". In: *2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI)*. IEEE. 2021, pp. 399–409.

[Fen+22] Tianyue Feng et al. "Multimodal critical-scenarios search method for test of autonomous vehicles". In: *Journal of intelligent and connected vehicles* 5.3 (2022), pp. 167–176.

[Gao+22] Feng Gao et al. “Performance Limit Evaluation Strategy for Automated Driving Systems”. In: *Automotive Innovation* 5.1 (2022), pp. 79–90.

[HSE22] Michael Hoss, Maike Scholtes, and Lutz Eckstein. “A review of testing object-based environment perception for safe automated driving”. In: *Automotive Innovation* 5.3 (2022), pp. 223–250.

[Li+22] Jia Li et al. “A domain-specific language for simulation-based testing of IoT edge-to-cloud solutions”. In: *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems*. 2022, pp. 367–378.

[Lou+22] Guannan Lou et al. “Testing of autonomous driving systems: where are we and where should we go?” In: *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 2022, pp. 31–43.

[Ope22] OpenAI. *ChatGPT: Optimizing Language Models for Dialogue*. 2022. URL: https://openai.com/blog/chatgpt/.

[Shi+22] Yunyang Shi et al. “An integrated traffic and vehicle co-simulation testing framework for connected and autonomous vehicles”. In: *IEEE Intelligent Transportation Systems Magazine* 14.6 (2022), pp. 26–40.

[Sun+22] Yang Sun et al. “LawBreaker: An approach for specifying traffic laws and fuzzing autonomous vehicles”. In: *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*. 2022, pp. 1–12.

[Tex22] Texas Department of Public Safety. *Texas DMV Handbook*. 2022. URL: https://driving-tests.org/texas/tx-dmv-drivers-handbook-manual/.

[Wan+22a] Jiangong Wang et al. “Parallel vision for long-tail regularization: Initial results from IVFC autonomous driving testing”. In: *IEEE Transactions on Intelligent Vehicles* 7.2 (2022), pp. 286–299.

[Wan+22b] Xinghua Wang et al. “Autonomous driving testing scenario generation based on in-depth vehicle-to-powered two-wheeler crash data in China”. In: *Accident Analysis & Prevention* 176 (2022), p. 106812.

[Wan+22c] Xuezhi Wang et al. “Self-consistency improves chain of thought reasoning in language models”. In: *arXiv preprint arXiv:2203.11171* (2022).

[Wei+22] Jason Wei et al. “Chain-of-thought prompting elicits reasoning in large language models”. In: *Advances in neural information processing systems* 35 (2022), pp. 24824–24837.

[Xu+22] Frank F Xu et al. “A systematic evaluation of large language models of code”. In: *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*. 2022, pp. 1–10.

[Zha+22] Peixing Zhang et al. “Performance evaluation method for automated driving system in logical scenario”. In: *Automotive Innovation* 5.3 (2022), pp. 299–310.

[Zho+22] Rui Zhou et al. “Genetic algorithm-based challenging scenarios generation for autonomous vehicle testing”. In: *IEEE Journal of Radio Frequency Identification* 6 (2022), pp. 928–933.

[CL23] Yue Cao and CS Lee. "Robot behavior-tree-based task generation with large language models". In: *arXiv preprint arXiv:2302.12927* (2023).

[Den+23] Yao Deng et al. "Target: Traffic rule-based test generation for autonomous driving systems". In: *arXiv preprint arXiv:2305.06018* (2023).

[GÖK23] Çağrı Güzay, Ege Özdemir, and Yahya Kara. "A Generative AI-driven Application: Use of Large Language Models for Traffic Scenario Generation". In: *2023 14th International Conference on Electrical and Electronics Engineering (ELECO)*. IEEE. 2023, pp. 1–6.

[Hu+23] Yihan Hu et al. "Planning-oriented autonomous driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 17853–17862.

[Las+23] Md Tahmid Rahman Laskar et al. "A systematic study and comprehensive evaluation of chatgpt on benchmark datasets". In: *arXiv preprint arXiv:2305.18486* (2023).

[LT23] Artem Lykov and Dzmitry Tsetserukou. "Llm-brain: Ai-driven fast generation of robot behaviour tree based on large language model". In: *arXiv preprint arXiv:2305.19352* (2023).

[MLI23] Antonio Valerio Miceli-Barone, Alex Lascarides, and Craig Innes. "Dialogue-based generation of self-driving simulation scenarios using Large Language Models". In: *arXiv preprint arXiv:2310.17372* (2023).

[Pat+23] Sagar Pathrudkar et al. "SAFR-AV: Safety Analysis of Autonomous Vehicles using Real World Data–An end-to-end solution for real world data driven scenario-based testing for pre-certification of AV stacks". In: *arXiv preprint arXiv:2302.14601* (2023).

[Sha+23] Hao Shao et al. "Safety-enhanced autonomous driving using interpretable sensor fusion transformer". In: *Conference on Robot Learning*. PMLR. 2023, pp. 726–737.

[Xi+23] Zhiheng Xi et al. "The rise and potential of large language model based agents: A survey". In: *arXiv preprint arXiv:2309.07864* (2023).

[Yin+23] Zhangyue Yin et al. "Do Large Language Models Know What They Don't Know?" In: *arXiv preprint arXiv:2305.18153* (2023).

[Zho+23] Ziyuan Zhong et al. "Language-guided traffic simulation via scene-level diffusion". In: *Conference on Robot Learning*. PMLR. 2023, pp. 144–177.

[C-N24] C-NCAP. *Appendix L: ACTIVE SAFETY ADAS TEST PROTOCOL [S]*. 2024.

[LKK24] Dong-Whan Lee, Tae-Lim Kim, and Seong-Jin Kwon. "A Study on the Driving Performance Analysis for Autonomous Vehicles Through the Real-Road Field Operational Test Platform". In: *International Journal of Precision Engineering and Manufacturing* (2024), pp. 1–11.

[Li+24] Quanyi Li et al. "ScenarioNet: Open-source platform for large-scale traffic scenario simulation and modeling". In: *Advances in neural information processing systems* 36 (2024).

[Liu+24] Haotian Liu et al. "Visual instruction tuning". In: *Advances in neural information processing systems* 36 (2024).

[You+24] Alex Young et al. “Yi: Open foundation models by 01. ai”. In: *arXiv preprint arXiv:2403.04652* (2024).

[Zha+24] Guosheng Zhao et al. “DriveDreamer-2: LLM-Enhanced World Models for Diverse Driving Video Generation”. In: *arXiv preprint arXiv:2403.06845* (2024).