Computing&AI Connect
2025, Vol. 1, Cite as: doi:10.x/journal.x.x.x

A Novel MLLM-based Approach for Autonomous Driving in Different Weather Conditions

SondaFourati, Wael Jaafar *, and Noura Baccar.

***Abstract*—Autonomous driving (AD) technology promises to revolutionize daily transportation by making it safer, more efficient, and more comfortable. Its role in reducing traffic accidents and improving mobility is vital to the future of intelligent transportation systems. AD systems (ADS) are expected to function reliably across diverse and challenging environments. However, existing solutions often struggle under harsh weather conditions such as foggy, rainy, or stormy conditions, and mostly rely on unimodal inputs, thus limiting their adaptability and performance. Meanwhile, multimodal large language models (MLLMs) have shown remarkable capabilities in perception, reasoning, and decision-making, yet their application in AD, particularly under extreme environmental conditions, remains largely unexplored. Consequently, this paper proposes MLLM-AD-4o, a novel AD agent that leverages prompt engineering to integrate camera and LiDAR inputs for enhanced perception and control. MLLM-AD-4o dynamically adapts to available sensor modalities and is built upon GPT-4o to ensure contextual reasoning and decision-making. To support realistic evaluation, the agent was developed using the LimSim++ framework, which integrates the SUMO and CARLA driving simulators. Experiments are conducted under harsh conditions, including bad weather, poor visibility, and complex traffic scenarios. The MLLM-AD-4o agent's robustness and performance are assessed for decision-making, perception, and control. The obtained results demonstrate the agent's ability to maintain high levels of safety and efficiency, even in extreme conditions, and using different perception components (e.g., cameras only, cameras with LiDAR, etc.). Finally, this work provides valuable insights into integrating MLLMs with AD frameworks, paving the way for fully safe ADS.**

***Index Terms*—Autonomous driving, multimodal large language models, harsh environment, CARLA, LimSim++, GPT-4o.**

I. Introduction

A. General Context

Recent advances that integrate blockchain, artificial intelligence (AI), and Internet-of-Things (IoT) technologies have substantially enhanced the design of secure and intelligent systems across a wide range of domains, including smart cities, industrial automation, and vehicular ecosystems, by ensuring data integrity, operational efficiency, and resilience [1]–[6]. Autonomous Driving Systems (ADS) are poised to become a cornerstone of next-generation transportation infrastructures. Their safe and efficient deployment critically depends on robust AI-driven frameworks, including those that synergize with IoT and blockchain technologies for real-time decision-making and secure data exchange [7].

S. Fourati and N. Baccar are with the Computer Systems Engineering Department, Mediterranean Institute of Technology (MedTech), Tunis, Tunisia. W. Jaafar is with the Software and IT Engineering Department, Ecole de Technologie Supérieure (ÉTS), Montreal, QC H3C 1K3, Canada. E-mails: {sonda.fourati, noura.baccar}@medtech.tn; wael.jaafar@etsmtl.ca.

Manuscript received XX XX, 20XX; .

Moreover, ADS are expected to operate safely and reliably in any environment, including urban, rural, and highway. However, real-world ADS deployment presents several challenges, particularly for adverse weather conditions such as rain, fog, and low-light, in which traditional ADS's perception and control systems struggle to operate. To address this, modern ADS rely typically on a combination of heterogeneous sensor modalities, e.g., cameras, LiDAR, and radar, to capture and interpret complex driving scenes.

To meet the growing demands for safety, efficiency, and adaptability in ADS, recent research has increasingly turned to AI-driven approaches, supported by large language models (LLMs) and multimodal LLMs (MLLMs) [8], [9]. The latter offers advanced perception, reasoning, and decision-making capabilities. Specifically, MLLMs combine multimodal data, including images, video, and audio data, with the advanced reasoning capabilities of LLMs [10]. Hence, they can act as decision-making agents for various applications, including medicine, education, and intelligent transport systems (ITS). MLLMs can provide accurate and timely responses to dynamic driving conditions by processing vast amounts of driving data to understand and predict complex traffic patterns, thus enabling autonomous vehicles (AVs) to learn from their environment and enhance their driving performance. In addition, MLLMs contribute significantly to the development of ADS by providing the computational power required to analyze the driving scene in real time and make accurate decisions [11].

In this context, various approaches have been proposed to use MLLMs for AD. Indeed, several strategies have been proposed based on prompt engineering [12]–[14], fine-tuning [15]–[19], and reinforcement learning with human feedback (RLHF) [20]–[22]. Furthermore, MLLMs toward AD provisioning could be utilized for perception and scene understanding [23]–[26], question answering [27], [28], planning and control [16], [29], and multitasking [30].

Despite their diverse methods and applications, MLLMs experience limitations in ADS. For instance, fine-tuning-based approaches are computationally expensive and may result in models overfitting for specific tasks or environments. Also, RLHF can align models with human preferences, however, it still faces scaling and generalization issues beyond particular training data instances. In contrast, MLLMs, particularly vision language models (VLMs) such as GPT-4o, have shown promising capabilities in tasks involving perception, reasoning, and decision-making. The latter can simultaneously integrate and interpret visual and textual information, enabling interactive and adaptive AI systems.

B. Motivation and Problem Statement

Despite the growing use of MLLMs in perception and decision-making, no existing approach integrates multimodal inputs (e.g., LiDAR and camera) using prompt engineering to enable robust autonomous driving in harsh environmental conditions. This work addresses this gap by proposing MLLM-AD-4o, a flexible sensor-adaptive AD agent. This work aims to address the following ADS issues:

- **Need for weather-robust AD:** Despite the rapid progress in ADS, robust perception and reasoning under adverse weather conditions remain an open issue. Accordingly, this work aims to close this gap by introducing an MLLM-based ADS system that is resilient to varying environmental conditions.
- **Multimodal adaptability and sensor awareness**: Real-world driving requires adaptability to available sensor inputs. A model that can reason effectively using LiDAR, front/rear cameras, or any combination thereof is necessary.
- **Harnessing the reasoning capabilities of MLLMs:** The ability of models like GPT-4o to perform complex reasoning and decision-making when properly prompted presents an opportunity to enhance ADS, especially via prompt engineering tailored for driving tasks.
- **Lack of open, reproducible frameworks for AD testing in harsh conditions:** Many existing benchmarks do not sufficiently simulate diverse weather and traffic conditions or provide extensibility. In contrast, this work extends LimSim++ to introduce a new benchmark within the CARLA simulator for reproducible testing in challenging weather environments.

C. Contributions and Paper Organization

To tackle the above challenges, this work introduces MLLM-AD-4o, a novel prompt engineering MLLM-based approach for AD, which is capable of handling autonomous driving in any weather condition. The proposed approach integrates different sensor modalities, including LiDAR and cameras, allowing the system to adapt according to available sensor data. Based on prompt engineering, MLLM-AD-4o provides enhanced environment perception, scene understanding, reasoning, and decision-making in ADS. Hence, our contributions can be summarized as follows:

- Unlike previous works, this contribution integrates for the first time the harsh environment driving conditions into the CARLA simulator by leveraging functions and modules from the LimSim++ framework. This work has been documented and disseminated in Github for open public access and reproducibility [31].
- The proposed MLLM-AD-4o is a novel MLLM-based AD agent that leverages GPT-4o [32] for driving perception and control decision-making.
- Through extensive experiments, a performance analysis of the proposed MLLM-AD-4o in terms of safety, comfort, efficiency, and speed score metrics, and under different weather conditions and types of involved sensor data (e.g., front and/or rear cameras and/or LiDAR) was conducted.
- The cumulative distribution functions (CDFs) are used for deeper insight into the agent's performance variability.

This combination of prompt engineering, sensor adaptability, and resilience to weather is the core contribution that sets this work apart from existing MLLM-based AD approaches.

The remainder of the paper is organized as follows. Section II, presents the related works that integrate LLMs/MLLMs into driving agents in a closed-loop using the CARLA simulator. Section III, explains the basic concepts of ADS and LLM/MLLMs with a focus on the GPT-4o architecture and functions. Section IV presents the architecture, main modules, and technical details for integrating an MLLM agent driver in a closed-loop environment using Limsim++. Section V evaluates the performance of MLLM-AD-4o in diverse scenarios and conditions. Finally, Section VI closes the paper.

D. List of Abbreviations

Below, we present the list of acronyms used in this paper.

AD	Autonomous Driving
ADS	Autonomous Driving Systems
API	Application Programming Interface
AV	Autonomous Vehicles
CDF	Cumulative Distribution Functions
DRL	Deep Reinforcement Learning
ITS	Intelligent Transport Systems
LanGen	Language Generator
LiDAR	Light Detection and Ranging
LLM	Large Language Models
MAE	Mean Absolute Error
MLLM	Multimodal Large Language Models
NLP	Natural Language Processing
SDK	Software Development Kit
TTC	Time-to-Conflict
QA	Question-Answering
VLM	Vision-Language Model

II. Related Work

Recently, there has been a significant effort to enhance the efficiency and reliability of ADS using LLMs and MLLMs. Among proposed driving agents, this work focuses on the related works based on prompt engineering. Authors of [33] proposed "LLM-driver", a framework integrating numeric vector modalities, into pre-trained LLMs for question-answering (QA). LLM-driver uses object-level 2D scene representations to fuse vector data into a pre-trained LLM with adapters. A language generator (lanGen) is used to ground vector representations into LLMs. The model's performance was evaluated on perception and action prediction using mean absolute error (MAE) for predictions, accuracy of traffic light detection, and normalized errors for acceleration, brake pressure, and steering. In [34], the SurrealDriver framework has been proposed. It consists of an LLM-based generative driving agent simulation framework with multitasking capabilities that integrate perception, decision-making, and control processes to

manage complex driving tasks. In [35], the authors developed LLM-based driver agents with reasoning and decision-making abilities, aligned with human driving styles. Their framework utilizes demonstrations and feedback to align the agents' behaviors with those of humans, utilizing data from human driving experiments and post-driving interviews. Also, the authors of [36] proposed the Co-driver framework for planning & control and trajectory prediction tasks. Co-driver utilizes prompt engineering to understand visual inputs and generate driving instructions, while it uses deep reinforcement learning (DRL) for planning and control. In [37], the authors introduced the PromptTrack framework, an approach to 3D object detection and tracking, by integrating cross-modal features within prompt reasoning. PromptTrack uses language prompts that act as semantic guides to enhance the contextual understanding of a scene. Finally, authors in [13] proposed HiLM-D as an efficient technique to incorporate high-resolution information in ADS for hazardous object localization.

Despite the variety of proposed driving agents, to our knowledge, no study has fully assessed the performance of MLLM-based driving agents within a closed-loop framework under harsh environmental conditions. Table I summarizes the aforementioned contributions with a comparison to this work.

III. Background

The key modules associated with an AV are *perception*, *planning*, *localization*, *decision-making*, and *action or control* as highlighted in Fig. 1. The *perception* module ensures sensing of the surroundings environment and identifies the driving scene. The main goal of the *localization* module is to estimate with high precision and accuracy the vehicle position on the map. *Path planning and decision* modules establish the waypoints that the vehicle should follow when moving through the surroundings. The set of waypoints corresponds to the vehicle trajectory. The *action and control* module deals with the system's actuation such as braking, steering, and acceleration.

Several MLLMs have been developed [9], with a subset of them suitable for ADS as discussed in Section II. In this work, the GPT-4o [32] was selected. GPT-4o is an optimized version of GPT-4 built around the latter's foundational transformer architecture with enhancements in understanding, generating, and maintaining context across interactions. GPT-4o includes various specialized modules to improve its operation, namely (i) language understanding module, (ii) language generation module, (iii) context management module, (iv) task-specific modules, and (v) optimization and fine-tuning modules. Programming GPT-4o involves using application programming interfaces (APIs) or software development kits (SDKs), custom prompts, or fine-tuning and customization. In our work, interaction with GPT-4o is realized via API calls, where a customized prompt to guide the model response is utilized.

IV. Materials and Methods: Proposed MLLM-Based Driving Agent in a Closed-Loop Environment

This section presents the tools used to integrate the proposed MLLM-based driving agent, a.k.a., MLLM-AD-4o.

A. LimSim++ Framework

LimSim++ is a closed-loop framework for deploying MLLMs for autonomous driving, where multimodal data, such as text, audio, and video, can be analyzed, and then accurately react to driving scenarios [40]. LimSim++ integrates algorithms for real-time decision-making, obstacle detection, and navigation, making it an interesting tool for designing, developing, testing, and refining ADS.

To do so, LimSim++ simulates a closed-loop system with specifications of the environment including road topology, dynamic traffic flow, navigation, and traffic control. The MLLM's agent is based on prompts that provide real-time scenario information in visuals or text. The MLLM-supported driving agent system can process information, use tools, develop strategies, and assess itself. Fig. 2 depicts the architecture of Limsim++ and explains its main modules as follows:

- *Simulation system*: It gathers the scenario information from SUMO and CARLA, and packages it as input for the MLLM, such as visual content, scenario cognition, and task description.
- *Driver Agent*: It communicates with the "simulation system" to make driving judgments and employs MLLMs to interpret them. Data is represented as prompts for MLLMs to make suitable driving decisions. The outputs of MLLMs influence vehicle behavior and are kept in the case log system for knowledge accumulation. Following the simulation, the decisions are evaluated as follows: Those that performed well are immediately added to the memory, while poor decisions are added after reflection. Building upon the results of the previous experiment, the reflection module refines the incorrect decision-making process by incorporating expert experience. The modified decision results are then added to the Driver Agent's memory modules, serving as driving experiences to provide few-shot instances for the agent in subsequent decision-making processes. This ensures continuous improvement and knowledge accumulation within the system.

The MLLM-AD-4o architecture was designed by customizing the above modules as follows:

1) The original "MPGUI.py" file within LimSim++ and its related dependencies are modified to include the visualization of six cameras (rather than the default three front cameras) in the autonomous driving simulation. Hence, a comprehensive view is provided, which is crucial to evaluating the perception capability of the driving system, analyzing/debugging the AV's sensor inputs, and decision-making.
2) The default "VLM-Driver-Agent" was updated to allow changes in the scenario and integrate new sensors and environmental conditions.
3) To automate the setup of different weather conditions, a novel function in the CARLA simulator was created .

TABLE I: Summary of related works

Ref.	Contribution	Used Models	Used Data	AD Tasks	Performance Metrics	Harsh Env.
[38]	Design and validation of an LLM Driver that interprets and reasons about driving situations to generate adequate actions.	GPT-3.5; RL agent trained with PPO, lanGen, and trainable LoRA modules.	160k QA driving pairs dataset; Control Commands Dataset.	Perception and action prediction.	Traffic light detection accuracy; Acceleration, brake, and steering errors.	No
[34]	LLM-based agent in a simulation framework to manage complex driving tasks.	GPT-3 and GPT-4.	Driving Behavior Data; Simulation data from CARLA simulator; NLP libraries.	Perception, control, and decision-making.	Collision rate.	No
[35]	LLM-based driver agents with reasoning and decision-making, aligned with human driving styles.	GPT-4.	Private dataset from a human driver; NLP library; RL library.	Behavioral alignment; Human-in-the-Loop system.	Collision rate, average speed, throttle percentage, and brake percentage.	No
[36]	Co-driver agent, based on a vision language model, for planning, control, and trajectory prediction.	Qwen-VL (9.6 billion parameters), including a visual encoder, a vision-language adapter, and the Qwen LLM.	CARLA simulation data; ROS2; Customized dataset.	Adjustable driving behaviors; Trajectory and lane prediction; Planning and control.	Fluctuations frequency, and running time.	Foggy/gloomy; Rainy/gloomy
[39]	PromptTrack framework for 3D object detection and tracking by integrating cross-modal features within prompt reasoning.	VoVNetV2 to extract visual features; RoBERTa to embed language prompts.	NuPrompt for 3D perception in AD.	3D object detection and tracking; Scene understanding.	Average multiple object tracking precision (AMOTA), and identity switches (IDS).	No
[13]	High-resolution scene understanding using proposed HiLM-D.	BLiP-2; Q-Former; MiniGPT-4.	DRAMA dataset; Pytorch; Visual Encoder; Query detection module; ST-adapter module.	Risky object detection; Vehicle intentions' and motions' prediction.	Mean intersection over union (mIoU) for detection; BLEU-4, METEOR, CIDER, and SPICE for captioning.	No
This work	Implementation of harsh environment scenarios and performance evaluation of an MLLM-based driving agent.	GPT-4o.	Dedicated Prompt.	Perception and decision-making.	Safety, comfort, efficiency, and speed scores.	Heavy rain; Storm; Foggy; Wetness; Good

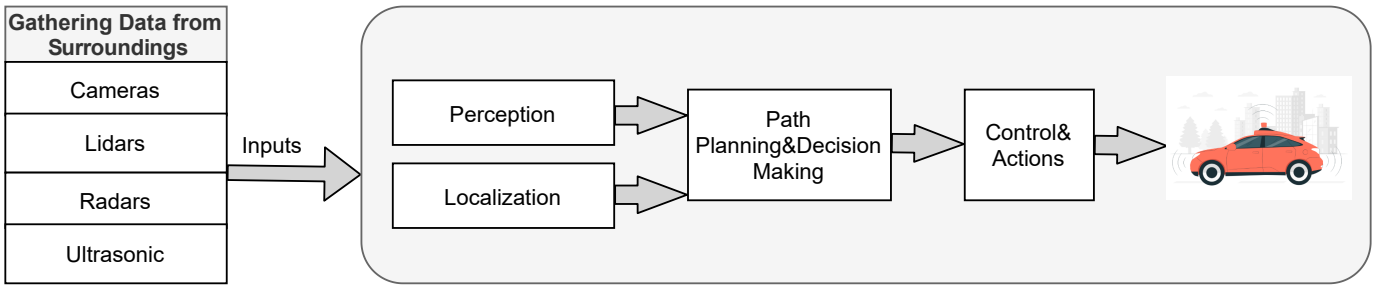


Fig. 1: Typical architecture of ADS.

V. Experimental Results

A. Definition of Metrics and Parameters

To accurately assess the AD performance of the proposed MLLM-AD-4o, the following performance metrics (called scores) are introduced:

- *Safety score:* The safety level is commonly assessed through Time-to-Conflict (TTC), which is a measure of the time remaining until a potential collision occurs between two moving objects, assuming no changes in their current trajectories. It serves as an indicator of the urgency to take corrective actions to avoid the conflict. When TTC falls below a specific threshold, a potential risk warrants penalties. The safety score can be given by

$$\text{Safety\_score} = \begin{cases} 1, & \text{if } \tau_e \geq \tau_{\text{th}} \\ \tau_e/\tau_{\text{th}}, & \text{otherwise,} \end{cases} \tag{1}$$

  where τ_e is the TTC of the AV and τ_{th} is the threshold value of TTC. Higher values reflect safer traveling.
- *Comfort score:* It evaluates the smoothness of the AV ride. Using empirical data, reference values can be determined for different driving styles, such as cautious, normal, and aggressive. It is computed as the average of the summation of acceleration score (acc_score), jerk score, lateral acceleration score (lat_acc_score), and lateral jerk score (lat_jerk_score), as shown below

$$\begin{aligned} \text{Comfort\_score} &= \frac{1}{4}\left(\text{acc\_score} + \text{jerk\_score}\right. \\ &+ \left.\text{lat\_acc\_score} + \text{lat\_jerk\_score}\right). \end{aligned} \tag{2}$$

  A higher comfort score means that traveling is smoother.
- *Efficiency score:* Driving efficiency is computed according to the vehicle's speed. In regular traffic conditions, the AV should maintain a speed at least equivalent to the average speed of surrounding vehicles. In sparse traffic conditions, the AV should approach the road's speed limit for efficiency. This score is computed as 3.

$$\text{Efficiency\_score} = \begin{cases} 1.0, & \text{if } v_e \geq v^* \\ \frac{v_e}{v^*}, & \text{otherwise,} \end{cases} \tag{3}$$

  where v_e represents the speed of the AV and $v^* \in \{v_{avg}, v_{limit}\}$ is the targeted speed for efficiency, being v_{avg} as the average speed of surrounding vehicles, and v_{limit} as the road's speed limit. A high value means efficient traveling.
- *Speed score:* The speed limit score (or speed score) penalizes the vehicle for exceeding the speed limit. This score is set to 0.9 when the AV exceeds the speed limit and 1 otherwise. This score is expressed as

$$\text{Speed\_score} = 1 - 0.1 \times \frac{\text{Nbr\_frames\_with\_speeding}}{\text{Total\_nbr\_frames}}, \tag{4}$$

  where "Nbr_frames_with_speeding" is the number of captured frames while speeding, and "Total_nbr_frames" is the total number of captured frames.

In the following simulations, we set up $\alpha_1 = \alpha_2 = 0.25$, $\alpha_3 = 0.5$, and $v_{limit} = 50$ km/h (i.e., 13.89 m/s).

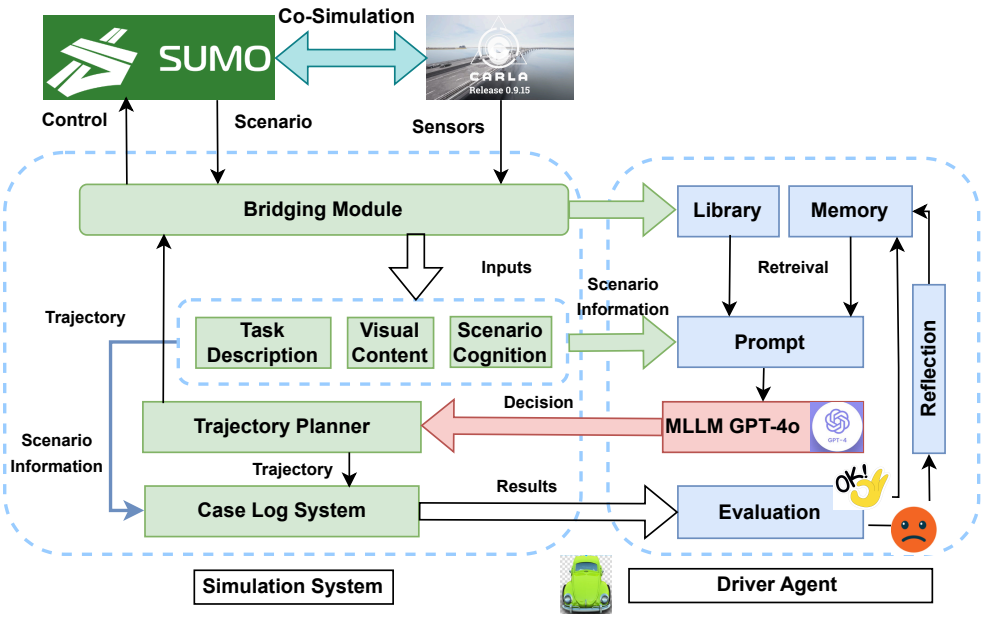


Fig. 2: Integration of MLLM-AD-4o in LimSim++ (adapted with permission from [40]).

B. Interaction with GPT-4o API

To use the GPT-4o API with prompts to process the driving environment, with data mainly collected from CARLA and SUMO, the following steps are conducted: [31]:

1) Install OpenAI package and get OpenAI API key.
2) Capture images from the front and back cameras (in CARLA), save them in a suitable format, and then convert them for API submission.
3) Prepare the API request, i.e., send the prompt with the image data to the GPT-4o API. The prompt describes what we want the model to do with the images and environmental data. Specifically, we ask the model to analyze the surroundings and provide a driving decision. Fig. 3 presents an example of used prompts in simulated scenarios that use six cameras. As illustrated, the prompts start with a clear role definition via the use "You are GPT-4o, a large multi-modal model..."; which enforces the model to behave like a decision-making assistant in autonomous driving, emphasizing responsibility, situational awareness, and accuracy. Then, the prompt explicitly defines the inputs (Front-view image, Back-view image, Navigation information, and a list of possible actions) and instructs GPT-4o to describe the visual scene, to reason about the next move using navigation data and visual inputs, and to decide on the next action. Finally, the prompt enforces the output format to be structured into three clear sections corresponding to *Description*, *Reasoning*, and *Decision*. This structure encourages explainability and transparency in the model's decisions. Moreover, the decision must match exactly one item in the action set, e.g., *Idle* (IDLE), *Acceleration* (AC), *Deceleration* (DEC), *Turn_Left* (TL), and *Turn_Right* (TR). This is critical to avoid any ambiguity and to reduce the number of consumed tokens.
4) GPT-4o API processes the gathered data (e.g., navigation info, lane info, vehicle info, and surrounding info) to make a decision, then sends a response to the MLLM agent in the form of a driving action.
5) The agent converts the action received from the GPT-4o API to behavior, then the trajectory planner module updates the path according to the decision made by the driver agent, to be sent to the CARLA system and executed. Fig. 4 below illustrates an example of GPT-4o's decision-making in a simulation under "Stormy weather". Accordingly, the decision taken is "Accelerate" despite the stormy weather with reduced visibility. When making this decision, the GPT-4o API considered the absence of immediately ahead or in adjacent lane obstacles in the processed images, as well as the vehicle's current speed and location (as described in the prompt). It concluded that acceleration is appropriate in this situation due to the lack of nearby hazards, the safe road surface visibility, and the need to maintain efficient traffic flow.

C. Setup of Weather Conditions

This work, considered, in addition to "Good weather", four harsh weather conditions, namely "Heavy rain", "Storm", "Foggy", and "Wetness". A new function in the CARLA simulator, named "set_weather", is developed to configure different weather conditions. This function has eight parameters:

- *Cloudiness:* The amount of cloud cover in the sky. A high value means more clouds.

Prompt:: in case of using Front and back cameras

You are GPT-4o, a large multi-modal model trained by OpenAI. Now you act as a mature driving assistant, who can give accurate and correct advice for human driver in complex urban driving scenarios. You'll receive some images from the onboard camera.
You'll need to make driving inferences and decisions based on the information in the images.
At each decision frame, you receive navigation information and a collection of actions.
You will perform scene description, and reasoning based on the navigation information, the front-view image and back-view image. Eventually you will select the appropriate action output from the action set.
Make sure that all of your reasoning is output in the `## Reasoning` section, and in the `## Decision` section you should only output the name of the action, e.g. `AC`, `IDLE` etc.

Your answer should follow this format:
Description
Your description of the front-view image, and your description of the back-view image.
Reasoning
reasoning based on the navigation information, the front-view image and the back-view image.
Decision
one of the actions in the action set.(SHOULD BE exactly same and no other words!)
"""

Fig. 3: Used prompt by GPT-4o (For AV with six cameras).

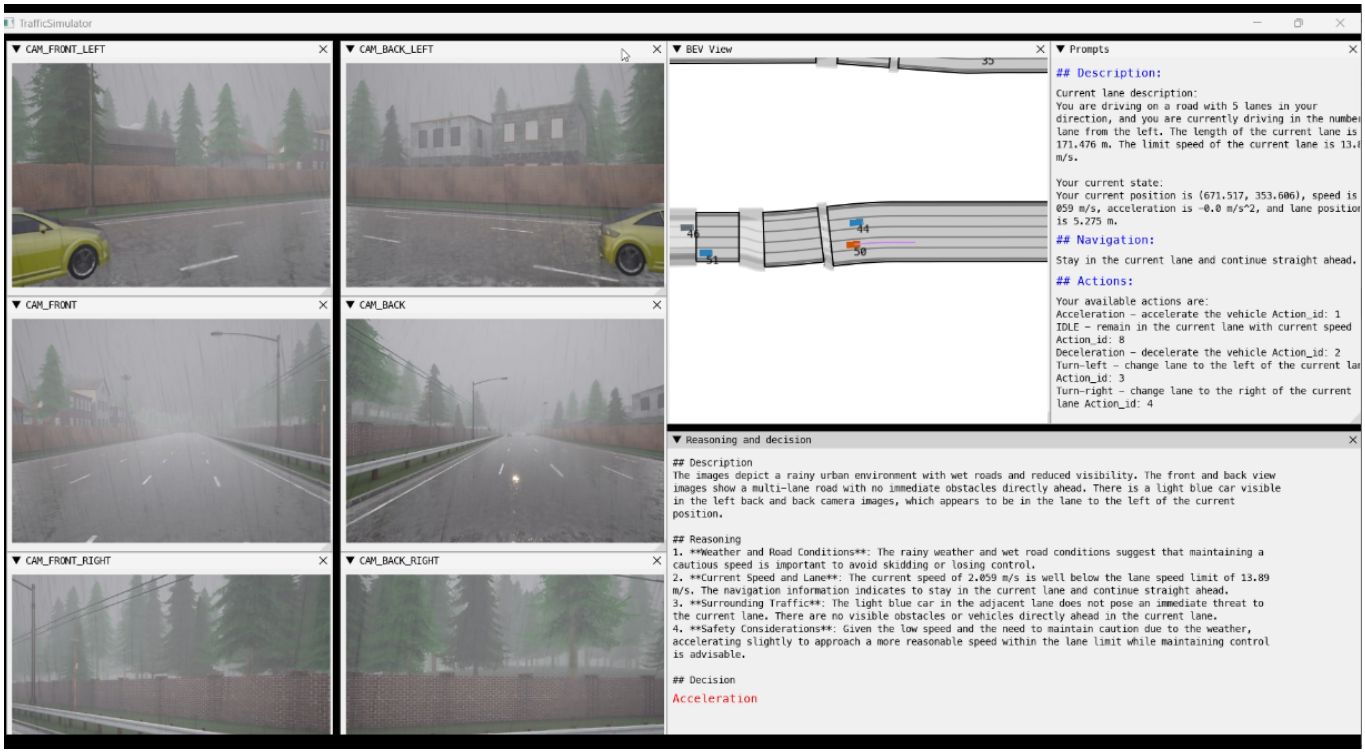


Fig. 4: Example of autonomous driving in stormy weather (For AV with six cameras).

- *Precipitation:* The intensity of rain. A high value means heavier rain.
- *Precipitation_deposits:* The amount of water accumulating on the ground due to precipitation. A high value means important accumulation.
- *Wind_intensity:* The strength of wind. A high value means stronger wind.
- *Sun_altitude_angle:* The angle of the sun above the horizon. Lower values can simulate dawn, dusk, or low-light conditions.
- *Fog_density:* The density of fog in the environment. A high value implies thicker fog.
- *Fog_distance:* The distance at which the fog starts to become noticeable.
- *Wetness:* The wetness of the road surfaces. A high value makes the roads wetter.

In Table II, we summarize the parameters' values to set up each weather condition using the function "set_weather" [31].

D. Integration of Semantic LiDAR in LimSim++

CARLA supports various sensing tools, including traditional LiDAR and semantic LiDAR. Traditional LiDAR provides raw distance measurements in the form of a point cloud,

TABLE II: Parameters to set up the weather conditions

Case	Set_weather parameters values
Heavy Rain	self.set_weather(cloudiness=80.0, precipitation=70.0, precipitation_deposits=60.0, wind_intensity=30.0, sun_altitude_angle=45.0, fog_density=10.0, fog_distance=10.0, wetness=80.0)
Storm	self.set_weather(cloudiness=80.0, precipitation=100.0, precipitation_deposits=100.0, wind_intensity=100.0, sun_altitude_angle=20.0, fog_density=20.0, fog_distance=10.0, wetness=80.0)
Fog	self.set_weather(cloudiness=40.0, precipitation=5.0, precipitation_deposits=5.0, wind_intensity=10.0, sun_altitude_angle=60.0, fog_density=70.0, fog_distance=3.0, wetness=10.0)
Wetness	self.set_weather(cloudiness=30.0, precipitation=0.0, precipitation_deposits=0.0, wind_intensity=0.0, sun_altitude_angle=70.0, fog_density=0.0, fog_distance=0.0, wetness=100.0)
Good Weather	self.set_weather(cloudiness=00.0, precipitation=00.0, precipitation_deposits=00.0, wind_intensity=00.0, sun_altitude_angle=60.0, fog_density=00.0, fog_distance=20.0, wetness=00.0)

with no inherent understanding of the objects in the scene. In contrast, semantic LiDAR provides both the point cloud and an understanding of what each point represents, thus simplifying tasks that involve scene understanding, object recognition, and data processing reduction. Semantic LiDAR is selected within this work. To add the latter in the perception environment within Limsim++, several steps should be followed [31]:

1) Attach "Semantic_LiDAR" function to the AV and gather data in CARLA.
2) Save and process LiDAR data.
3) Add "Semantic_LiDAR" data to the prompt request.

E. Simulation Results and Discussion

This section assesses the performance of the proposed MLLM-AD-4o autonomous driving agent in different conditions. Each experiment is built with the same scenario of our AV driving between points A and B, where it travels on multilane main roads and has to maneuver in curves. The autonomous driving agent has to take one decision among five possible decisions in each time frame, including idle (no change in current behavior), acceleration, deceleration, turn left, or turn right.

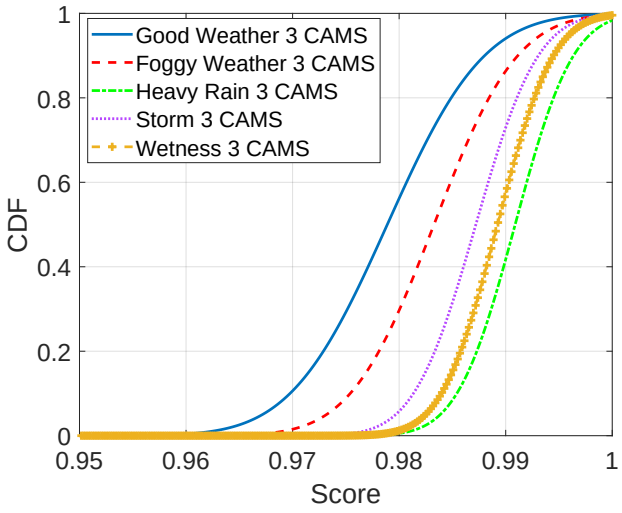
Figs. 5a-5d present the cumulative distribution functions (CDFs) for the safety, comfort, efficiency, and speed scores/-metrics, respectively, under various weather conditions, when 3 cameras are used for autonomous driving by the proposed MLLM-AD-4o method[1]. Fig. 5a, illustrates when the weather is good (blue line), the agent takes a riskier behavior that may cause collisions than in a harsh condition. This also impacts the passengers' comfort during travel as shown in Fig. 5b. Nevertheless, in good weather, the driving behavior is the most efficient one as shown in Fig. 5c and it aligns with the speed performance depicted in Fig. 5d, This trade-off highlights that while the AV may be driven more aggressively and with less comfort, its efficiency improves as it aligns more closely with the surrounding traffic's speed. In contrast, in harsh weather conditions, e.g., heavy rain (green line) or wetness (dark yellow line), safety is enforced through slower driving, which translates into a better comfort score, offering a smoother and more cautious ride, at the expense of degraded efficiency and speed performances.

[1]Fig. 5d illustrates also speed score results for the 6 cameras case, which will be discussed later.

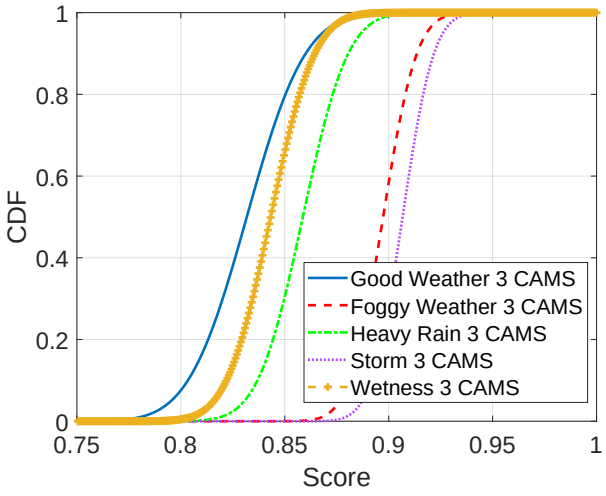
Among the weather conditions, we notice in Fig. 5 that the best safety CDF is achieved in heavy rain (green line), the best comfort CDF in stormy weather (pink line), the best efficiency CDF in good weather, and the best speed CDF in all of the foggy, heavy rain, and stormy weather conditions (red triangle). This suggests that the MLLM-AD-4o driving agent considers heavy rain the most unsafe environment requiring low-speed traveling. Also, traveling can be smooth in stormy weather, while driving is most efficient in good weather. Finally, in foggy/heavy rain/storm conditions, the agent is more likely to respect the speed limits as it favors slower driving to guarantee security, which is not the case in good and wet weather conditions. These results reflect the trade-offs the MLLM-AD-4o agent makes between various performance scores, under different weather patterns.

Figs. 6a-6c, depict the CDFs of the safety, comfort, and efficiency scores when 6 cameras (3 front + 3 rear) are used by the AD agent. In good weather, MLLM-AD-4o achieves now the best safety CDF performance (blue triangle), at the expense of comfort and efficiency. Indeed, the additional information brought by the 3 rear cameras makes the agent more precautious regarding the behavior of the surrounding vehicles, which was not possible when it only relied on 3 cameras. This is confirmed with the speed CDF result (purple line) in Fig. 5d which is higher than that of the 3 cameras' CDF. According to Fig. 6b, the best CDFs are realized in stormy/foggy weather (with a preference for stormy) at the expense of efficiency and safety. Indeed, in these situations, the AD agent is more likely to drive smoothly with minor behavior changes in response to poor visibility, with a higher safety risk and reduced efficiency, due to the misalignment with the surrounding road traffic behavior. Moreover, the best efficiency CDF is obtained in wet conditions, as shown in Fig. 6c. Indeed, with the knowledge of its 6 cameras, the AV maintains a speed closer to the average of other vehicles by continuously adapting to the dynamics of the road and surrounding traffic. This is validated through the results in Fig. 5d where the 6 cameras' speed CDF is better than the 3 cameras' one in wetness. However, as illustrated in Figs. 6a-6b, this behavior negatively impacts the safety and comfort scores. In summary, the addition of the rear cameras enhances the vehicle's ability to perceive and respond to potential collision threats, particularly in challenging weather conditions.

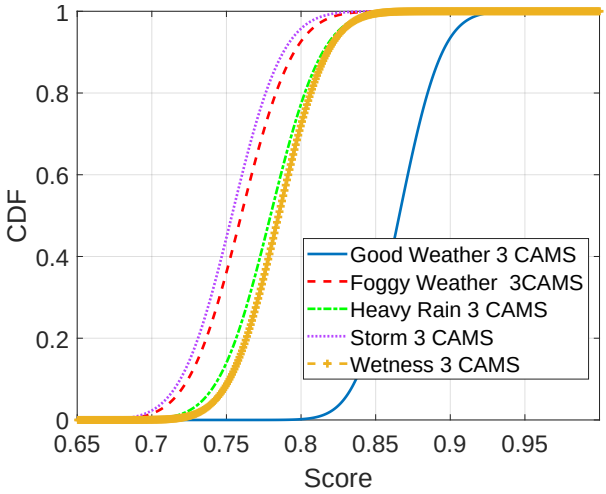
Figs. 7a-c illustrate the CDFs of the safety, comfort, and efficiency scores, when the AD agent is equipped with 3 (front) cameras only, 6 (3 front + 3 rear) cameras only, 3 (front) cameras + LiDAR, and 6 (3 front + 3 rear) cameras + LiDAR, respectively, operating in a heavy rain condition. The best safety CDF is obtained when the LiDAR is used with 3 cameras (green triangle in Fig. 7a). Indeed, activating the LiDAR in adverse weather conditions enhances the system's awareness of its environment leading to more responsible driving actions, and thus to an improved safety. Nevertheless, safety and efficiency CDFs degrade when LiDAR is activated with 6 cameras. This suggests that rear cameras' information might distort the agent's comprehension of its surroundings when combined with LiDAR. The best comfort score is also



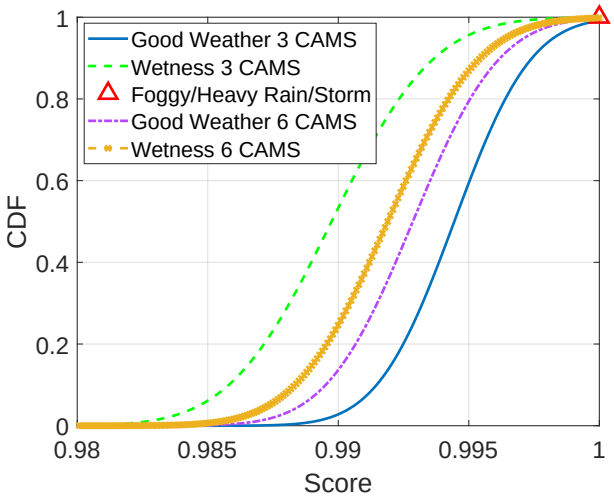
(a) CDF of safety score



(b) CDF of comfort score

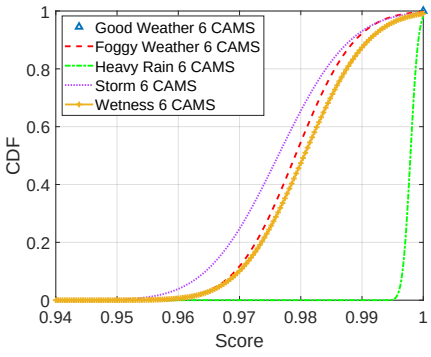


(c) CDF of efficiency score

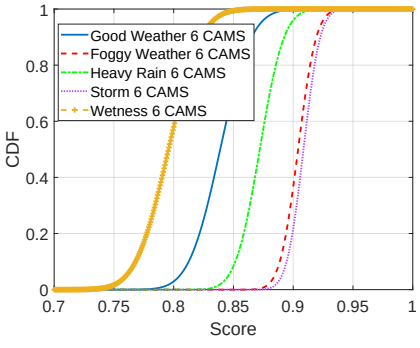


(d) CDF of speed score

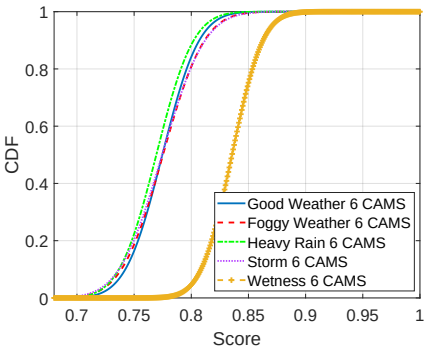
Fig. 5: CDFs of MLLM-AD-4o scores (Front cameras).



(a) CDF of safety score

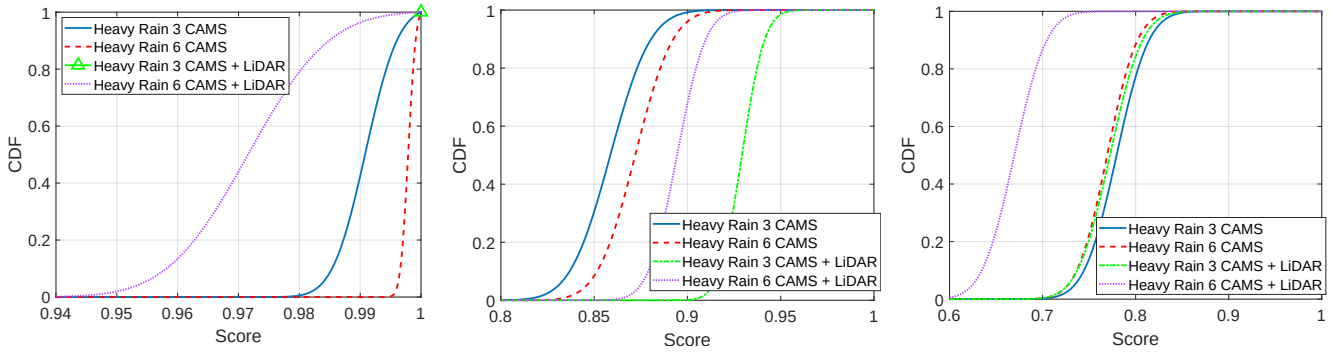


(b) CDF of comfort score



(c) CDF of efficiency score

Fig. 6: CDFs of MLLM-AD-4o scores (Front+rear cameras).



(a) CDF of safety score (b) CDF of comfort score (c) CDF of efficiency score

Fig. 7: CDFs of MLLM-AD-4o scores (Combinations of cameras with LiDAR; Heavy rain).

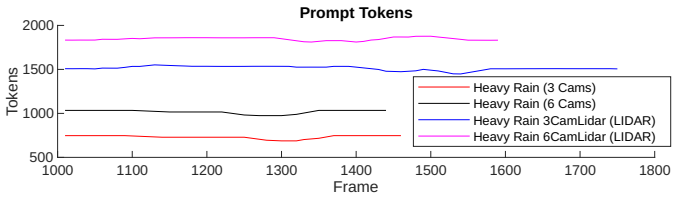


Fig. 8: Comparison of token load (different sensor settings, heavy rain).

achieved when 3 cameras are activated with LiDAR (green line in Fig. 7b), while the best efficiency score is realized with a 3-camera system (blue line in Fig. 7c). Nevertheless, the activation of LiDAR with the 3 cameras (green line in Fig. 7c) realizes similar efficiency CDF performance to the latter. Based on these results, the activation of LiDAR with cameras should be triggered in a timely manner, in particular conditions, and with specific configurations, as the full combination of cameras with LiDAR does not necessarily lead to better AD performances.

To quantitatively assess the scalability and latency implications of querying GPT-4o under different sensor configurations, we plot in Fig. 8 the prompt token load per frame for different sensor configurations (3 cameras, 6 cameras, 3 cameras plus LiDAR, and 6 cameras plus LiDAR) in heavy rain weather. As shown, the prompt size increases with the number of sensors. Specifically, configurations that include LiDAR contribute significantly to the token count, reaching around 1850 tokens for the "6Cam+LiDAR" setup, while simpler configurations, e.g., "3 Cams", consume under 800 tokens. This trend illustrates a linear scalability pattern in prompt size with sensor complexity, which affects the API querying time. Given that OpenAI's GPT-4o API latency is partially a function of input prompt size, these findings allow us to quantify a trade-off between perception fidelity and real-time responsiveness. For example, under a 30 frame-per-second (FPS) rate, using "6Cam+LiDAR" could saturate the API token limits or introduce delays depending on the network bandwidth and batching strategies.

To further explore the latency trade-off in a realistic low-bandwidth setting, we conduct an additional experiment to evaluate the token usage and inference time when querying GPT-4o through its API. The results, summarized in Fig. 9, present four subplots capturing prompt tokens (top-left), completion tokens (top-right), total tokens (bottom-left), and total response time (bottom-right) for 3-camera and 6-camera sensor configurations. We notice that increasing the number of cameras leads to higher prompt and completion token counts, which correlate with increased total token counts. Nevertheless, the total response time is lower with 6-camera data, probably due to the model producing shorter completions, because of a richer initial context. These results emphasize the nuanced trade-off between richer multimodal context and token budget constraints, a key consideration for real-time deployment of LLM-based autonomous driving agents.

VI. Conclusion

This work introduced MLLM-AD-4o, a novel AD agent based on GPT-4o, capable of performing accurate AV perception and safe control decisions via prompt engineering across diverse environmental conditions. Harsh weather scenarios were integrated into the CARLA simulator through the LimSim++ framework, enabling the evaluation of MLLM-AD-4o with different sensor configurations. For clarity, the performances have been assessed in terms of safety, comfort, efficiency, and speed score CDFs. The obtained results prove that combining front and rear cameras' data significantly improves performance compared to single-view configurations, in clear weather. Moreover, integrating LiDAR with a moderate number of cameras (e.g., only three) yields optimal performance in different weather conditions. In contrast, excessive sensor inputs, e.g., from LiDAR and six cameras simultaneously, can be counterproductive. These findings demonstrate the practicality of MLLM-AD-4o for AD and the importance of context-aware sensor configuration. In future work, we will expand our framework to more complex traffic scenarios and explore the use of lighter MLLMs that are better adapted for real-time decision-making.

AUTHOR CONTRIBUTIONS

S. Fourati: Conceptualization, methodology, software, investigation, data curation, writing, original draft preparation; W. Jaafar: Validation, Formal analysis, writing, review, and editing, supervision, project administration; N. Baccar: Formal analysis, Visualization, writing, review, and editing, supervision.

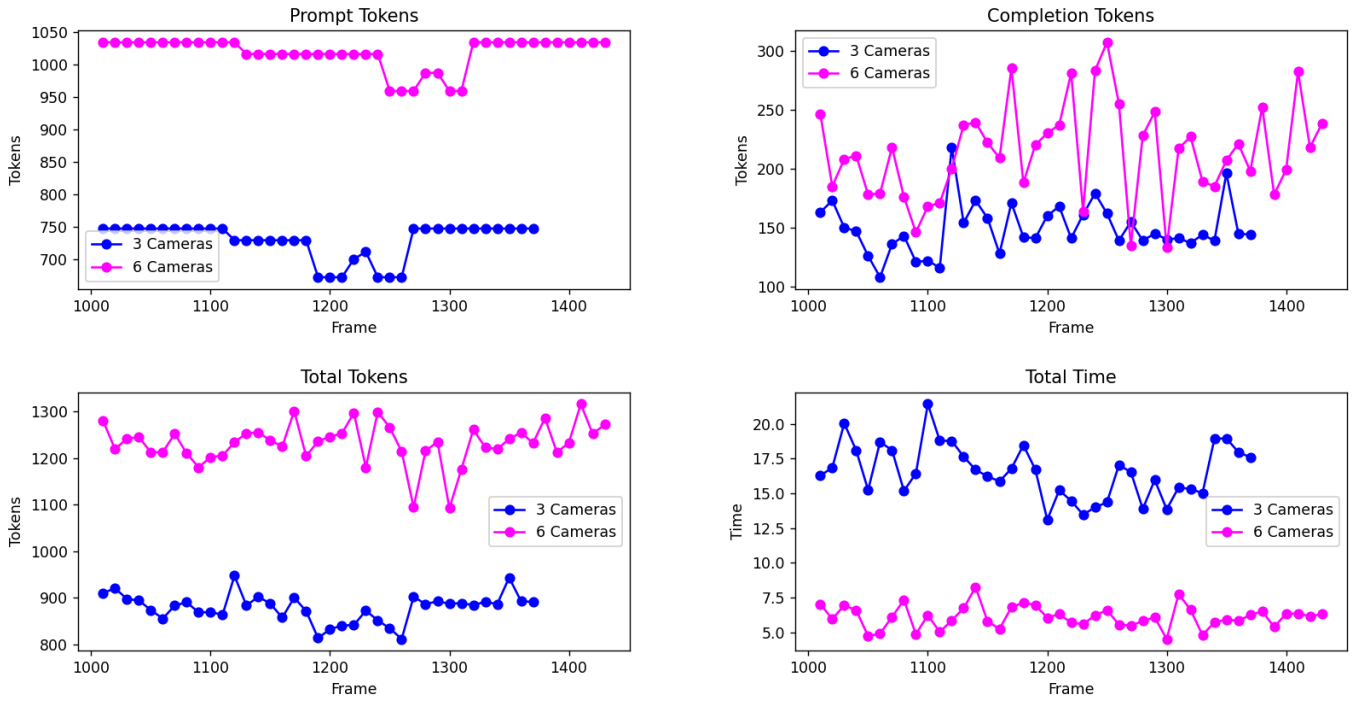


Fig. 9: Comparison of prompt tokens (top-left), completion tokens (top-right), total tokens (bottom-left), and total response time per frame (bottom-right) when using 3 or 6 cameras.

AVAILABILITY OF DATA AND MATERIALS

The data used in this study were generated through extensive simulations using the LimSim++ tool. The simulation outputs, including traffic scenarios, sensor data, and decision logs, are available from the corresponding author upon reasonable request.

CONFLICTS OF INTEREST

The authors have no competing interests to declare relevant to this article's content. All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

Funding

This work is funded in part by a Mitacs Globalink Research Award scholarship and in part by NSERC Discovery Grant RGPIN-2023-03757.

ACKNOWLEDGEMENTS

The authors want to acknowledge the Mediterranean Institute of Technology (MedTech), and École de Technologie Supérieure (ÉTS) for providing support in this research.

REFERENCES

[1] A. A. Khan, A. A. Laghari, A. M. Baqasah, R. Bacarra, R. Alroobaea, M. Alsafyani, and J. A. J. Alsayaydeh, "BDLT-IoMT—a novel architecture: SVM machine learning for robust and secure data processing in internet of medical things with blockchain cybersecurity," *The J. of Supercomput.*, vol. 81, no. 1, pp. 1–22, 2025.

[2] A. A. Khan, J. Yang, A. A. Laghari, A. M. Baqasah, R. Alroobaea, C. S. Ku, R. Alizadehsani, U. R. Acharya, and L. Y. Por, "BAIoT-EMS: Consortium network for small-medium enterprises management system with blockchain and augmented intelligence of things," *Engineer. Appl. of Artifi. Intelli.*, vol. 141, p. 109838, Feb. 2025.

[3] A. A. Khan, A. A. Laghari, S. A. Inam, S. Ullah, and L. Nadeem, "A review on artificial intelligence thermal fluids and the integration of energy conservation with blockchain technology," *Discover Sustain.*, vol. 6, no. 1, pp. 1–18, 2025.

[4] A. A. Khan, A. A. Laghari, A. M. Baqasah, R. Alroobaea, T. R. Gadekallu, G. A. Sampedro, and Y. Zhu, "ORAN-B5G: A next generation open radio access network architecture with machine learning for beyond 5G in industrial 5.0," *IEEE Trans. Green Commun. Network.*, 2024.

[5] A. A. Khan, A. A. Laghari, R. Alroobaea, A. M. Baqasah, M. Alsafyani, R. Bacarra, and J. A. J. Alsayaydeh, "Secure remote sensing data with blockchain distributed ledger technology: a solution for smart cities," *IEEE Access*, 2024.

[6] A. A. Khan, S. Dhabi, J. Yang, W. Alhakami, S. Bourouis, and L. Yee, "B-LPoET: A middleware lightweight proof-of-elapsed time (PoET) for efficient distributed transaction execution and security on blockchain using multithreading technology," *Comput. Elec. Engineer.*, vol. 118, p. 109343, Aug. 2024.

[7] A. A. Khan, J. Yang, S. A. Awan, A. M. Baqasah, R. Alroobaea, Y.-L. Chen, C. S. Ku, and L. Y. Por, "Artificial intelligence, internet of things, and blockchain empowering future vehicular developments: A comprehensive multi-hierarchical lifecycle review," *Human-centric Comput. Info. Sci.*, vol. 15, no. 13, 2025.

[8] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang *et al.*, "A survey on evaluation of large language models," *ACM Trans. Intelli. Syst. Technol.*, vol. 15, no. 3, pp. 1–45, 2024.

[9] S. Fourati, W. Jaafar, N. Baccar, and S. Alfattani, "XLM for autonomous driving systems: A comprehensive review," *arXiv preprint arXiv:2409.10484*, 2024.

[10] Y. Wang, W. Chen, X. Han, X. Lin, H. Zhao, Y. Liu, B. Zhai, J. Yuan, Q. You, and H. Yang, "Exploring the reasoning abilities of multimodal large language models (MLLMs): A comprehensive survey on emerging trends in multimodal reasoning," *arXiv preprint arXiv:2401.06805*, 2024.

[11] C. Cui, Y. Ma, X. Cao, W. Ye, Y. Zhou, K. Liang, J. Chen, J. Lu, Z. Yang, K.-D. Liao *et al.*, "A survey on multimodal large language models for autonomous driving," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2024, pp. 958–979.

[12] L. Wen, D. Fu, X. Li, X. Cai, T. Ma, P. Cai, M. Dou, B. Shi, L. He, and Y. Qiao, "Dilu: A knowledge-driven approach to autonomous driving with large language models," *arXiv preprint arXiv:2309.16292*, 2023.

[13] X. Ding, J. Han, H. Xu, W. Zhang, and X. Li, "HiLM-D: Towards high-resolution understanding in multimodal large language models for autonomous driving," *arXiv preprint arXiv:2309.05186*, 2023.

[14] J. Yuan, S. Sun, D. Omeiza, B. Zhao, P. Newman, L. Kunze, and M. Gadd, "RAG-Driver: Generalisable driving explanations with retrieval-augmented in-context learning in multi-modal large language model," in *Proc. Robot.: Sci. and Syst. (RSS) Conf.*, 2024, pp. 1–14.

[15] T. Wang, E. Xie, R. Chu, Z. Li, and P. Luo, "DriveCoT: Integrating chain-of-thought reasoning with end-to-end driving," *arXiv preprint arXiv:2403.16996*, 2024.

[16] W. Wang, J. Xie, C. Hu, H. Zou, J. Fan, W. Tong, Y. Wen, S. Wu, H. Deng, Z. Li *et al.*, "DriveMLM: Aligning multi-modal large language models with behavioral planning states for autonomous driving," *arXiv preprint arXiv:2312.09245*, 2023.

[17] Y. Huang, J. Sansom, Z. Ma, F. Gervits, and J. Chai, "DriVLMe: exploring foundation models as autonomous driving agents that perceive, communicate, and navigate," *Proc. Vis. and Langua. for Autonom. Driv. and Roboti. Wrkshp*, 2024.

[18] G. Liao, J. Li, and X. Ye, "VLM2Scene: Self-supervised image-text-LiDAR learning with foundation models for autonomous driving scene understanding," in *Proc. AAAI Conf. on Artifi. Intelli.*, vol. 38, no. 4, 2024, pp. 3351–3359.

[19] S. Wang, Z. Yu, X. Jiang, S. Lan, M. Shi, N. Chang, J. Kautz, Y. Li, and J. M. Alvarez, "OmniDrive: A holistic LLM-agent framework for autonomous driving with 3D perception, reasoning and planning," *arXiv preprint arXiv:2405.01533*, 2024.

[20] H. Tian, K. Reddy, Y. Feng, M. Quddus, Y. Demiris, and P. Angeloudis, "Enhancing autonomous vehicle training with language model integration and critical scenario generation," *arXiv preprint arXiv:2404.08570*, 2024.

[21] J. Mao, J. Ye, Y. Qian, M. Pavone, and Y. Wang, "A language agent for autonomous driving," in *Proc. Conf. Langua. Model. (COMS)*, 2024.

[22] J. Zhang, Z. Huang, A. Ray, and E. Ohn-Bar, "Feedback-guided autonomous driving," in *Proc. IEEE/CVF Conf. Comp. Vis. Patt. Recogn.*, 2024, pp. 15 000–15 011.

[23] H. I. Ashqar, T. I. Alhadidi, M. Elhenawy, and N. O. Khanfar, "Leveraging multimodal large language models (MLLMs) for enhanced object detection and scene understanding in thermal images for autonomous driving systems," *Automat.*, vol. 5, no. 4, pp. 508–526, 2024.

[24] S. Luo, W. Chen, W. Tian, R. Liu, L. Hou, X. Zhang, H. Shen, R. Wu, S. Geng, Y. Zhou *et al.*, "Delving into multi-modal multi-task foundation models for road scene understanding: From learning paradigm perspectives," *IEEE Trans. Intelli. Veh. (Early Access)*, pp. 1–25, 2024.

[25] R. Li, Z. Zhang, C. He, Z. Ma, V. M. Patel, and L. Zhang, "Dense multimodal alignment for open-vocabulary 3D scene understanding," in *Europ. Conf. Comp. Vis.* Springer, 2024, pp. 416–434.

[26] C. Zhong, S. Cheng, M. Kasoar, and R. Arcucci, "Reduced-order digital twin and latent data assimilation for global wildfire prediction," *Natural haz. Earth Syst. Sci.*, vol. 23, no. 5, pp. 1755–1768, 2023.

[27] Z. Xu, Y. Zhang, E. Xie, Z. Zhao, Y. Guo, K.-Y. K. Wong, Z. Li, and H. Zhao, "DriveGPT4: Interpretable end-to-end autonomous driving via large language model," *IEEE Robot. Automa. Lett.*, vol. 9, no. 10, pp. 8186–8193, 2024.

[28] S. Cheng, H. Chassagnon, M. Kasoar, Y. Guo, and R. Arcucci, "Deep learning surrogate models of jules-inferno for wildfire prediction on a global scale," *IEEE Trans. Emerg. Top. Computa. Intelli.*, vol. 9, no. 1, pp. 444–454, 2025.

[29] C. Cui, Y. Ma, X. Cao, W. Ye, and Z. Wang, "Receive, reason, and react: Drive as you say, with large language models in autonomous vehicles," *IEEE Intelli. Transport. Syst. Mag.*, vol. 16, no. 4, pp. 81–94, 2024.

[30] Z. Li, W. Wang, Y. Cai, X. Qi, P. Wang, D. Zhang, H. Song, B. Jiang, Z. Huang, and T. Wang, "UnifiedMLLM: Enabling unified representation for multi-modal multi-tasks with large language model," *arXiv preprint arXiv:2408.02503*, 2024.

[31] Fourati, Sonda. (2024) MLLM applied to autonomous driving across various weather conditions. Accessed: 2024-11-10. [Online]. Available: https://github.com/SondaFourati/MLLM-applied-to-autonomous-driving-across-various-weather-conditions/tree/main

[32] R. Islam and O. M. Moushi, "GPT-4o: The cutting-edge advancement in multimodal LLM," *Authorea Preprints*, 2024.

[33] L. Chen, O. Sinavski, J. Hünermann, A. Karnsund, A. J. Willmott, D. Birch, D. Maund, and J. Shotton, "Driving with LLMs: Fusing object-level vector modality for explainable autonomous driving," in *Proc. IEEE Int. Conf. Robo. Automa. (ICRA)*. IEEE, 2024, pp. 14 093–14 100.

[34] Y. Jin, R. Yang, Z. Yi, X. Shen, H. Peng, X. Liu, J. Qin, J. Li, J. Xie, P. Gao, G. Zhou, and J. Gong, "SurrealDriver: Designing LLM-powered generative driver agent framework based on human drivers' driving-thinking data," in *Proc. IEEE/RSJ Int. Conf. Intelli. Robo. Syst. (IROS)*, 2024, pp. 966–971.

[35] R. Yang, X. Zhang, A. Fernandez-Laaksonen, X. Ding, and J. Gong, "Driving style alignment for LLM-powered driver agent," in *Proc. IEEE/RSJ Int. Conf. Intelli. Robo. Syst. (IROS)*. IEEE, 2024, pp. 11 318–11 324.

[36] Z. Guo, A. Lykov, Z. Yagudin, M. Konenkov, and D. Tsetserukou, "Co-driver: VLM-based autonomous driving assistant with human-like behavior and understanding for complex road scenes," *arXiv e-prints*, pp. arXiv–2405, 2024.

[37] D. Wu, W. Han, Y. Liu, T. Wang, C.-z. Xu, X. Zhang, and J. Shen, "Language prompt for autonomous driving," in *Proc. AAAI Conf. Artifi. Intelli.*, vol. 39, no. 8, 2025, pp. 8359–8367.

[38] L. Chen, O. Sinavski, J. Hünermann, A. Karnsund, A. J. Willmott, D. Birch, D. Maund, and J. Shotton, "Driving with LLMs: Fusing object-level vector modality for explainable autonomous driving," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2024, pp. 14 093–14 100.

[39] D. Wu, W. Han, T. Wang, Y. Liu, X. Zhang, and J. Shen, "Language prompt for autonomous driving," *arXiv preprint arXiv:2309.04379*, 2023.

[40] D. Fu, W. Lei, L. Wen, P. Cai, S. Mao, M. Dou, B. Shi, and Y. Qiao, "Limsim++: A closed-loop platform for deploying multimodal LLMs in autonomous driving," in *Proc. IEEE Intelli. Veh. Symp. (IV)*. IEEE, 2024, pp. 1084–1090.