Shengheng Yan

Autonomous Driving Systems with Large Language Models: A Comparative Study of Interpretability and Motion Planning

Master's Thesis in Information Technology

May 16, 2024

University of Jyväskylä

Faculty of Information Technology

Author: Shengheng Yan

Contact information: `yansh@jyu.fi`

Supervisors: Tsvetomila Mihaylova, and Terziyan Vagan

Title: Autonomous Driving Systems with Large Language Models: A Comparative Study of Interpretability and Motion Planning

Työn nimi: Autonomiset ajoneuvojärjestelmät suurilla kielimalleilla: Vertailututkimus tulkittavuudesta ja liikkeen suunnittelusta

Project: Master's Thesis

Study line: Artificial Intelligence

Page count: 73+3

Abstract: In this master's thesis, we investigate the integration of large language models into autonomous driving systems, with a particular emphasis on their potential to enhance interpretability, decision-making, and planning capabilities. We implement both data-driven and knowledge-driven models within the CARLA simulator across diverse scenarios, focusing specifically on the TransFuser and LMDrive frameworks. This study provides a comparative analysis of these models utilizing a range of metrics. The results indicate that while LMDrive exhibits certain limitations in motion planning, it demonstrates significant competence in interpretability, particularly in recognizing traffic light signals and detecting bumpy road conditions.

Keywords: Autonomous Driving, Large Language Models, Interpretability, Deep Learning

Suomenkielinen tiivistelmä: Tässä pro gradu -tutkielmassa tutkimme suurten kielimallien integrointia autonomisiin ajoneuvojärjestelmiin, erityisesti niiden potentiaalia parantaa tulkittavuutta, päätöksentekoa ja suunnittelukyvykkyyttä. Toteutamme sekä datalähtöisiä että tietämyslähtöisiä malleja CARLA-simulaattorissa erilaisissa skenaarioissa keskittyen erityisesti TransFuser- ja LMDrive-kehyksiin. Tämä tutkimus tarjoaa vertailevan analyysin näistä malleista käyttäen useita mittareita. Tulokset osoittavat, että vaikka LMDrive osoittaa tiettyjä rajoituksia liikkeen suunnittelussa, se osoittaa merkittävää osaamista tulkittavuudessa, erityisesti liikennevalojen tunnistamisessa ja epätasaisen tien havaitsemisessa.

Avainsanat: Autonominen ajaminen, suuret kielimallit, tulkittavuus, syväoppiminen

Glossary

FC	Fully Connected Layer.	13
GRU	Gated Recurrent Unit.	16
LMDrive	Closed-Loop End-to-End Driving with Large Language Models.	3
MLP	Multilayer Perceptron.	14
MMFT	Multi-modal Fusion Transformer.	25
TD	Transformer Decoder.	20
TE	Transformer Encoder.	20
TransFuser	Imitation with Transformer-Based Sensor Fusion for Autonomous Driving.	3
VE	Vision Encoder.	26
ViT	Vision Transformer.	21

List of Figures

Figure 1. Fully Connected Layer ... 14
Figure 2. Multilayer Perceptron with 2 fully connected layers ... 15
Figure 3. The Architecture of GRU(N. Li et al. May 2021) ... 16
Figure 4. Transformer(Vaswani et al. 2023) ... 18
Figure 5. ViT(Dosovitskiy et al. 2021) ... 21
Figure 6. Architecture of TransFuser(Chitta et al. 2022) ... 24
Figure 7. Architecture of LMDrive(Shao et al. 2023) ... 26
Figure 8. A typical simulation process(Law and Kelton 1991) ... 29
Figure 9. Ego-vehicle start at yellow point, end at orange point follow pre-designed routes along with white line ... 34
Figure 10. The view of 3 conditions of weather ... 35
Figure 11. The T-intersection is ahead of the agent and the system instructs to turn right ... 45
Figure 12. The system instructs the agent to change course to the other lane ... 46
Figure 13. The system instructs the agent to drive straight ... 47
Figure 14. The system gives a notification about the yellow light ahead ... 49
Figure 15. The system asks for attention to the rugged road ahead ... 49
Figure 16. Notice: Attention is required for the rugged road surface head ... 50

List of Tables

Table 1. Prompt Engineering Approaches ... 9
Table 2. Fine-tuning Pre-trained Models ... 12
Table 3. Classification of different testing scenarios (Gold et al. June 2017) ... 34
Table 4. Configurations of Conflict Scenarios ... 36
Table 5. Configurations of TransFuser and LMDrive ... 38
Table 6. Results of TransFuser on 3 routes with 3 times evaluation ... 40
Table 7. Results of LMDrive on 3 routes with 3 times evaluation ... 41
Table 8. Results of different models in default weather ... 42
Table 9. Results of different models in conflict scenarios ... 43
Table 10. Samples of failure of instructions ... 44
Table 11. Notifications collected from 3 routes both on default and extreme conflict. ... 48
Table 12. Samples of Notification for Bumpy Road ... 50

Contents

1 INTRODUCTION ... 1

2 AUTONOMOUS DRIVING ... 4
- 2.1 Rule-based Methods to Data-driven Models ... 5
- 2.2 Knowledge-driven Models ... 6
  - 2.2.1 Prompt Engineering ... 6
  - 2.2.2 Fine-tuning Pre-trained Models ... 9
- 2.3 Summary ... 12

3 DEEP LEARNING ... 13
- 3.1 Traditional Neural Network Architectures ... 13
  - 3.1.1 Fully Connected Layer ... 13
  - 3.1.2 Multilayer Perceptron ... 14
  - 3.1.3 The GRU Network ... 16
- 3.2 Transformer ... 17
  - 3.2.1 Self-Attention Mechanism ... 18
  - 3.2.2 Multi-Head Attention ... 19
  - 3.2.3 Transformer Encoder and Decoder ... 20
- 3.3 Vision Transformer ... 21
  - 3.3.1 Input Embedding ... 21
  - 3.3.2 Positional Encoding ... 22
  - 3.3.3 Transformer Encoder ... 22
  - 3.3.4 Classification Head ... 23
- 3.4 Summary ... 23

4 DEEP LEARNING MODELS IN AUTONOMOUS DRIVING ... 24
- 4.1 TransFuser ... 24
  - 4.1.1 Multi-Modal Fusion Transformer ... 25
  - 4.1.2 Auto-Regressive Waypoint Prediction Network ... 25
- 4.2 LMDrive ... 26
  - 4.2.1 Vision Encoder ... 26
  - 4.2.2 Large Language Model ... 27
- 4.3 Summary ... 28

5 METHODOLOGY ... 29

6 EXPERIMENTS ... 32
- 6.1 CARLA Simulation ... 32
  - 6.1.1 Scenarios ... 33
  - 6.1.2 Conflict Scenarios ... 34
- 6.2 Autonomous Agents ... 36
- 6.3 Metrics ... 38

7 RESULTS ... 39

7.1 Comparison of TransFuser and LMDrive 39
    7.1.1 No Conflicts Scenarios 39
    7.1.2 Conflict Scenarios 42
7.2 Interpretability of LMDrive 43
    7.2.1 Failure of Instructions 43
    7.2.2 Traffic Lights Notifications 47
    7.2.3 Bumpy Road Detection 50
7.3 Summary 51

8 CONCLUSIONS 52

BIBLIOGRAPHY 53

APPENDICES 67

1 Introduction

The market trend in electric vehicles (EVs) is increasing rapidly and is predicted to reach 145 million sales by the year 2030 (Patil 2020). In the meantime, autonomous driving(AD) technology could enhance the convenience and efficiency of EVs even further. Such technology has seen significant progress, mainly driven by ongoing improvements in sensor technology such as millimeter-wave radar, light detection and ranging(LiDAR), cameras, etc. (Li and Ibanez-Guzman 2020; Xiang et al. 2023), and its integration of artificial intelligence (AI) (Chen, Li, et al. 2023).

Despite significant progress of AI in autonomous driving, ongoing challenges persist. A heavy dependence on data-driven methods leaves systems vulnerable to data bias, often leading to overfitting training data (Cao et al. 2023; Huang et al. 2023). This challenge hinders the capability of current autonomous driving systems to effectively tackle long-tail and cross-domain issues (Xiang et al. 2023; J. Wang et al. 2022), which restricts their adaptability in unfamiliar environments. Additionally, these systems suffer from a lack of interpretability (Zablocki et al. 2022; Zhang et al. 2023). Overall, **data-driven** methods rely on large datasets to train models for pattern recognition and prediction but can suffer from overfitting, lack generalization, and limited interpretability, while **knowledge-driven methods** integrate knowledge, common sense, and reasoning of human to enhance handling complex scenarios, improve interpretability, and enable more adaptive decision-making (Wang, Yang, and Wu 2023).

Notably, the term explainability is frequently associated with the concept of interpretability(Zablocki et al. 2022). **Interpretability** refers to the extent to which a human can comprehend the rationale behind a decision made by an AI system, with an emphasis on the clarity of the explanation (Miller 2018), while **explainability** includes interpretability and completeness, ensuring the explanation is both understandable and thoroughly covers all relevant aspects of the decision-making process. In this study, we concentrate on the interpretability.

Therefore, the need to explain the behaviors of autonomous driving is multifaceted (Zablocki et al. 2022). For example, **1. Debugging**: Explanations can aid engineers and researchers in enhancing future iterations by offering insights into edge cases, pitfalls, and possible failure points (Y. Tian et al. 2018; Hecker et al. 2020). **2. Trusting**: Explanations become essential for building user trust and facilitating the implementation of the technology when the system's performance matches human performance (Lee and Moray 1992; Choi and Ji July 2015; Shen et al. 2022; Zhang, Yang, and Robert 2020). **3. Improving**: If autonomous driving models significantly surpass human abilities in the future, explanations could help educate humans on improved driving techniques and better decision-making through machine teaching (Aodha et al. 2018).

Recent works, LM-Nav (Shah et al. 2022) and CLIP-MC (Jain et al. 2022) harness large-scale vision language pre-training to acquire linguistic knowledge from textual instructions and visual features from images by leveraging CLIP (Radford et al. 2021). These studies demonstrate the efficacy of this pre-trained model, presenting a compelling framework for addressing intricate navigation challenges using multimodal models (Chen, Wu, et al. 2023). Although large language models (LLMs) like GPT-3 (Brown et al. 2020) show the ability to handle complex linguistic instructions, the integration of LLMs and the field of autonomous driving remains unclear (Chen, Wu, et al. 2023).

While the majority of studies prioritize enhancing trajectory planning ability and developing new models, there is less emphasis on evaluating existing models and their potential to improve interpretability. Without thorough evaluation, the field may continue to advance technologically without addressing these crucial aspects. This could result in the development of autonomous systems that are less transparent, harder to trust, and potentially more error-prone. Moreover, the opportunity to refine and perfect existing technologies might be lost, leading to continual reinvention rather than focused improvement and optimization. Therefore, we propose four research questions to address this research gap:

1. **What existing approaches and large language models are available for developing knowledge-driven autonomous driving systems?**

   This question aims to survey and review existing resources and techniques in the field of autonomous driving. We focus on those that leverage large language model approaches. Understanding the current landscape of methodologies, datasets, and language models will provide a foundational understanding necessary for advancing research or application development in this area.

2. **How can a large language model be implemented in practice for motion planning in a simulation environment?**

   This question seeks to explore the practical implementation of large language models for motion planning, and driving interpretation, particularly within a simulated setting. The goal is to determine the feasibility, requirements, techniques, and potential barriers or efficiencies in integrating language models into autonomous driving systems.

3. **How does the performance of knowledge-driven models compare to data-driven models?**

   The objective is to assess and compare the effectiveness and efficiency of knowledge-driven models with more traditional data-driven models. This comparison could reveal insights into the advantages and disadvantages of knowledge-driven models.

4. **What is the effectiveness of the interpretability provided by knowledge-driven models?**

   This question addresses the need to assess how well knowledge-driven models can explain their decisions and actions. It is a crucial aspect for regulatory approval, user trust, and debugging purposes in autonomous systems. Understanding the level of interpretability can help improve model design and interface for better human-machine interaction.

The thesis structure is organized as follows: Chapter 2 introduces autonomous driving technology and its associated research, particularly in the context of deep learning models. Chapter 3 provides an overview of deep learning, establishing the foundation for the subsequent discussion of the TransFuser and LMDrive models, which are elaborated in Chapter 4. Chapter 5 justifies the simulation methodology employed. In Chapters 6 and 7, the experimental setup and findings are presented. Lastly, Chapter 8 offers conclusions and outlines potential future work.

2 Autonomous Driving

An Autonomous Driving Vehicle also referred to as a self-driving vehicle (Taeihagh and Lim July 2018; Thrun April 2010), is a vehicle capable of functioning with minimal or no human input (S. Xie et al. 2022). These vehicles are accountable for all driving tasks, including environmental perception, system monitoring, and vehicle control, which encompasses navigation from the point of origin to the intended destination (S. Xie et al. 2023).

A recent technical report by the National Highway Traffic Safety Administration (NHTSA) attributes 94% of road accidents to human error (Singh 2015). In light of this, Autonomous Driving vehicles are being developed with the potential to prevent accidents, reduce emissions, provide mobility for those with impairments, and alleviate driving-related stress (Crayton and Meier 2017).

Autonomous Driving (AD) has five to six levels, from manual to fully automatic, according to the definitions from the SAE (S.O.-R.A.V.S. Committee 2021), NHTSA (Blanco et al. August 2015), and German BASt (Tom M. Gasser August 2012). **Level 0** (No Driving Automation): The human driver is responsible for performing all driving tasks without any assistance from the vehicle's automated systems. **Level 1** (Driver Assistance): At this level, the vehicle is equipped with single or multiple systems that can assist the driver with either steering, acceleration, or braking. **Level 2** (Partial Driving Automation): The vehicle can control both steering and acceleration/deceleration, but the human driver must continuously monitor the driving environment and be prepared to take control at any moment. **Level 3** (Conditional Driving Automation): Vehicles at this level are capable of managing all driving tasks under certain conditions, but the human driver must be ready to intervene when the system requests. **Level 4** (High Driving Automation): The vehicle can handle all driving tasks and monitor the driving environment in specific scenarios without the need for driver attention, though human intervention may still be required in other situations. **Level 5** (Full Driving Automation): Vehicles operating at this level are fully autonomous and can perform all driving tasks in all conditions without any human intervention. Given technical constraints that delay the realization of fully autonomous driving to a distant future, our research is primarily focused on Level 3.

2.1 Rule-based Methods to Data-driven Models

Initially, path planning in autonomous driving often relied on **Rule-based** algorithms including A* search (Dolgov et al. 2008), which are based on a set of predefined, explicit instructions or rules that control the behavior of the vehicle. This method was particularly effective for problems with well-defined state spaces and deterministic environments.

As the field progressed, the complexity of autonomous driving tasks and environments led to a shift toward reinforcement learning(RL), and deep reinforcement learning (Udugama 2023; Tammewar et al. 2023). The agent's learning and decision-making process in **Data-driven** systems is driven by the goal of maximizing the cumulative reward, which is dynamically computed based on the agent's actions and the state of the environment. Most recently in 2021, a pivotal advancement was made in autonomous driving technology. With the availability of various sensor configurations that fit within a reasonable computational budget, the focus shifted towards integrating more modalities and sophisticated architectures such as Transformer (Vaswani et al. 2023). This approach aimed to enhance the global context and feature representation capabilities, as seen in TransFuser (Prakash, Chitta, and Geiger 2021; Chitta et al. 2022).

At the same time, the interpretability of autonomous driving systems has emerged as a key area of research interest (Hu et al. 2023). While earlier, small-scale language models such as initial versions of BERT (Devlin et al. 2019a) and GPT (Brown et al. 2020) have been utilized to process extensive driving data, their generalization abilities often fall short of optimal performance. More recently, advanced large language models (Agarwal et al. March 2023; Touvron, Martin, et al. 2023) have shown exceptional capabilities in context comprehension, response generation, and complex task management. These models are increasingly being combined with multi-modal systems (Brohan et al. 2023; Liu, Li, Wu, et al. 2023), facilitating a unified representation across different data types such as text, images, videos, and point clouds. This integration considerably boosts the system's ability to generalize and adapt swiftly to new situations, either without prior specific training (zero-shot) or with minimal training (few-shot).

2.2 Knowledge-driven Models

Differing from rule-based or data-driven approaches, knowledge-driven systems incorporate structured symbolic reasoning, enabling them to handle complex decision-making tasks that require abstract and logical reasoning (Wang, Yang, and Wu 2023). Knowledge-driven methods seek to incorporate information about driving scenarios into a knowledge-augmented representation space and infer it into a generalized driving semantic space (Wang, Yang, and Wu 2023). For example, knowledge-driven autonomous driving not only provides the vehicle with an understanding and reasoning of the real world but also enables it to navigate numerous challenging driving situations and adapt to dynamic environments. Therefore, the development of an interpretable knowledge-driven system and efficient end-to-end autonomous driving system has emerged as a focal point of research interest (Chen, Wu, et al. 2023).

Within this field, the related research can be classified into two primary categories based on the tuning of the large language model(Z. Yang et al. 2023): prompt engineering and fine-tuning pre-trained models.

2.2.1 Prompt Engineering

The advent of ChatGPT (OpenAI 2023) has spurred interest in LLMs through prompt engineering to enhance autonomous driving systems.

“**A Safety Perspective**”(Y. Wang et al. 2024) investigates how LLMs can be utilized as intelligent decision-makers in autonomous driving (AD), particularly in behavioral planning. They introduce two approaches: 1. Adaptive LLM-Conditioned Model Predictive Control (MPC): LLMs are integrated into the AD system to formulate constraints for a low-level MPC that handles trajectory planning. 2. Interactive Behavior Planning Scheme: The scheme uses a state machine where LLMs act as behavior planners and interactively manage behavioral states and transitions. They use GPT-4 as their LLM and are tested on HighwayEnv(Leurent 2018). The results show that their approach improves safety and efficiency while reducing latency compared to recent advanced methods like **Drive Like A Human** (Fu et al. 2023).

“**ChatGPT as Your Vehicle Co-Pilot**”(S. Wang et al. 2023) manages human-vehicle interaction, organizing information via a memory mechanism. They proposed Expert Oriented Black-Box tuning approach for optimization without retraining the LLMs(GPT-3.5-turbo). The Co-Pilot is tested through a joint simulation of Simulink(Documentation 2020) and CarSim(Johansson et al. 2004). The results show the Co-Pilot provides plausible maneuvers that meet human demands, such as safety, stability, or urgency.

Talk2Drive(Cui et al. 2024) presents a framework that translates verbal commands into control sequences for autonomous vehicles while personalizing driving experiences through a memory module. Their approach uses LLMs (GPT-4) to interpret spoken commands and execute driving decisions. A memory module archives past commands, actions, and feedback to refine preferences over time. Then Real-world tests are conducted in three key driving environments: highways, intersections, and parking lots. The result of the Talk2Drive framework shows that reducing driver takeovers by up to 78.8% while maintaining high-performance scores.

“**Receive, Reason, and React**” (Cui et al. 2023) proposes a novel framework to utilize LLMs in autonomous vehicles to enhance decision-making, provide transparent reasoning, and offer personalized driving experiences. The approach applies Chain-of-Thought Prompting(Wei et al. 2022) to LLMs. and the LLMs receive verbal commands, query relevant tools for context, and produce actionable driving instructions. A memory module also stores historical actions and preferences. Experiments are conducted using the HighwayEnv(Leurent 2018) simulator to assess how well LLMs like GPT-4 can interpret verbal commands and make informed decisions. The research demonstrates that integrating LLMs into autonomous vehicles improves their ability to reason, personalize responses, and make safe driving decisions.

Talk2BEV (Choudhary et al. 2023) introduces a framework that enhances Bird’s Eye View (BEV) maps with vision-language features, enabling comprehensive scene understanding and reasoning for autonomous driving tasks. Their approach starts with Generating BEV maps from vehicle sensors, Creating language-enhanced maps by using large vision language models(LVLMs) such as BLIP-2(J. Li et al. 2023), MiniGPT-4(Zhu et al. 2023), and InstructBLIP-2(Dai et al. 2023) to describe objects and align them with image-language features, and Answering user queries with LVLMs using context from these language-enhanced maps. The qualitative results demonstrate robust performance in predicting potential risky driving maneuvers through interactive dialogue.

SurrealDriver(Jin et al. 2023) seeks to tackle the challenges of simulating realistic human-like driving behaviors in urban environments such as CARLA(Dosovitskiy et al. 2017) and employs GPT-4. Their approach includes a DriverAgent responsible for perception, decision-making, and control, and a CoachAgent to provide driving guidelines based on driver interviews. The results indicate that implementing the framework with safety criteria reduced the collision rate by 57.46%, incorporating short-term memory lowered collision rates by 82.96%, and overall safety criteria, short-term memory, and long-term guidelines reduced collisions by 81.04% and improved human-likeness by 50%.

"**Drive a Like Human**"(Fu et al. 2023) explores the potential of using GPT-3.5 in a closed-loop driving system, and compares it to a traditional approach such as reinforcement learning. They measure the three key abilities (Reasoning, Interpretation, and Memorization) of a large language model. The results from the HighwayEnv simulation show that GPT-3.5 can use common sense to think and drive like humans.

LanguageMPC(Sha et al. 2023) uses ChatGPT as the center console (brain) for autonomous driving to analyze and reason about various scenarios. They implement Reinforcement Learning-based planning (RL), Model Predictive Control (MPC), and MPC with large language models and evaluate the approaches using the IdSim(Fitzgibbons et al. 2004).

DriveVLM (X. Tian et al. 2024) uses Qwen-VL(Bai et al. 2023) as the backbone of the large vision language model and integrates it with prompts into DriveVLM which takes sequences of images as Vision Language Model input and outputs the results of Chain-of-Thought, prompt engineering that comprises scene description, analysis, and hierarchical planning. They test the model with nuScenes(Caesar et al. 2020) dataset and the Scene Understanding for Planning(SUP-AD) dataset created by authors to evaluate scene description/analysis and a predefined set of decision-making options.

DiLu(Wen et al. 2024) explores how to empower the capabilities of large language models into autonomous driving systems. They evaluated their results on HighwayEnv with 20 experiences obtained from the lane-4-density-2 environment, and conducted experiments in the lane-5-density-3 setting, testing the closed-loop performance of 3-shot and 5-shot respectively. Also, they use CitySim to test the ability to use real-world datasets. The limitation they mention is 5-10 seconds of latency in the process of decision-making.

Name	LVM	Evaluation	Simulator
"**A Safety Perspective**"(Y. Wang et al. 2024)	GPT-4	1 scenario	HighwayEnv
"**ChatGPT as Your Vehicle Co-Pilot**"(S. Wang et al. 2023)	GPT-3.5	2 tasks	Simulink, CarSim
Talk2Drive(Cui et al. 2024)	GPT-4	3 scenarios	Real-world test
"**Receive, Reason, and React**"(Cui et al. 2023)	GPT-4	2 scenarios	HighwayEnv
Talk2BEV(Choudhary et al. 2023)	BLIP-2, MiniGPT-4	Talk2BEV-Bench	-
SurrealDriver(Jin et al. 2023)	GPT-4	Town10	CARLA
"**Drive a Like Human**"(Fu et al. 2023)	GPT3.5	2 scenarios	HighwayEnv
LanguageMPC(Sha et al. 2023)	GPT3.5	25 scenarios	IdSim
DriveVLM(X. Tian et al. 2024)	Qwen-VL	SUP-AD,nuScenes	-
DiLu(Wen et al. 2024)	GPT3.5	3 lane settings	HighwayEnv,CitySim

Table 1: Prompt Engineering Approaches

2.2.2 Fine-tuning Pre-trained Models

Compared to prompt engineering, a fine-tuning technique performs better in specific domains with targeted training data. The implementation of fine-tuning requires more computational resources than prompting.

DriveMLM(W. Wang et al. 2023) employs a multi-modal LLM like LLaMA-7B(Touvron, Lavril, et al. 2023) integrated with a Vision Transformer (ViT)(Dosovitskiy et al. 2021) from EVA-CLIP(Fang et al. 2022) for visual processing and a Sparse Pyramid Transformer (SPT)(H. Yang et al. 2023) for LiDAR data. They use CARLA to evaluate the model in different driving scenarios with varying weather, lighting, and environments. The result shows that DriveMLM achieves an overall decision prediction accuracy of 75.23%, significantly outperforming other models like LLaVA 1.5(Liu, Li, Li, et al. 2023) (22.92%) and InstructBLIP (17.92%) in Open-Loop Evaluation. The model obtains a Driving Score (DS) of 76.1 on the CARLA Town05 Long benchmark, surpassing the Apollo baseline by 4.7 points in

Closed-Loop Driving.

GPT-Driver (Mao, Qian, et al. 2023) introduces an approach to motion planning for autonomous vehicles using the OpenAI's GPT-3.5 model which reinterprets motion planning as a language modeling task by translating the inputs and outputs of the planner into linguistic tokens. It allows for describing trajectory coordinates and its reasoning process in natural language. The approach is evaluated on the nuScenes dataset. The approach has its limitations, for example, the model size and inference time, which could hinder its application in real-time autonomous driving situations. They suggest exploring the techniques of the smaller large language model. The other limitation is concerned with their evaluation, which addresses open-loop motion planning and does not fully account for error accumulation during the driving process. The closed-loop motion planning is mentioned for evaluating the performance of motion planners.

"**Driving with LLMs**" (Chen, Sinavski, et al. 2023) proposes an Object-Level Multimodal LLM framework, which aligns quantitative vector modalities with static large language model representations using vector captioning language data. They present a new dataset containing 160k question-answer (QA) pairs produced by GPT-3.5 derived from 10k driving scenarios collected by a reinforcement learning agent. GPT Grading and human grading are used for analyzing the performance of the model.

DriveGPT4 (Xu et al. 2024) is built upon the LLaMA architecture, and trained using a mix-finetuning strategy, which pre-training and fintuning with visual instruction tuning dataset created by the author's team. The dataset contains 56K video-text instruction-following samples and comprises 16K BDD-X(Kim et al. 2018) question-answering pairs and 40K QAs generated by ChatGPT. Their evaluation is focused on speed and turning angle, utilizing various metrics from the NLP community, such as CIDEr(Vedantam, Zitnick, and Parikh 2015), BLEU4(Papineni et al. 2002), and ROUGE-L(Lin July 2004).

Moreover, **LMDrive** (Shao et al. 2023) deals with the limitations of current autonomous driving systems when encountering long-tail unforeseen events. They introduce a dataset with about 64K data in 2 to 20-second clips which include navigation and notice instructions, multi-view sensor data from multiple modalities, and control signals. Their evaluation uses closed-loop experiments, which demonstrate LMDrive's effectiveness.

Agent-Driver (Mao, Ye, et al. 2023) uses GPT-3.5 as the primary LLM. The method utilizes chain-of-thought reasoning, including the identification of key objects and events, the formulation of high-level task plans, the generation of motion trajectories, and the performance of collision checks for self-assessment. Agent-Driver was evaluated using the large-scale nuScenes benchmark dataset. The results show strong few-shot learning by achieving superior performance with only a fraction of the training data, showing rapid adaptability. It also offers a transparent and understandable decision-making process by providing textual justifications.

DriveLM (Sima et al. 2023) fine-tuned BLIP-2 (J. Li et al. 2023) using the DriveLM-nuScene dataset, which is derived from nuScenes. The DriveLM-CARLA dataset was gathered utilizing CARLA version 0.9.14 as part of the Leaderboard 2.0 framework. They evaluated their model, DriveLM-Agent, on DriveLM-Data, which comprises DriveLM-nuScene and DriveLM-CARLA, using custom metrics: DriveLM-Metric. This is a standard metric from the nuScenes and Waymo(Sun et al. 2020) benchmarks

MTD-GPT(Liu, Hang, et al. 2023) uses GPT-2 architecture for multi-task decision-making. The model is trained in a custom-built simulation to forecast the next action by utilizing historical state-action-reward sequences and future rewards. The MTD-GPT model is tested in simulation scenarios on various tasks at unsignalized intersections and achieves comparable or superior performance compared to single-task RL models.

Name	LVM	Evaluation	Simulator
DriveMLM(W. Wang et al. 2023)	LLaMA-7B	nuScenes	-
GPT-Driver(Mao, Qian, et al. 2023)	GPT3.5	nuScenes	-
"**Driving with LLMs**"(Chen, Sinavski, et al. 2023)	LLaMA-7b	DataQA	Custom
DriveGPT4(Xu et al. 2024)	LLaMA-2	BDD-X	-
LMDrive(Shao et al. 2023)	LLaMa-2	LangAuto Benchmark	CARLA
Agent-Driver(Mao, Ye, et al. 2023)	GPT-3.5	nuScenes	-
DriveLM(Sima et al. 2023)	BLIP-2	DriveLM-Data	-
MTD-GPT(Liu, Hang, et al. 2023)	GPT-2	3 cases	Custom

Table 2: Fine-tuning Pre-trained Models

2.3 Summary

We conduct a review on the development and evolution of autonomous driving technologies, particularly focusing on how LLMs and data-driven approaches are improving the performance and reliability of these systems.

Overview of Autonomous Vehicles: Autonomous vehicles (AVs) are designed to navigate with minimal human intervention. According to the National Highway Traffic Safety Administration's findings, human error causes 94% of road accidents. Autonomous vehicles are expected to reduce accidents, improve accessibility, and decrease emissions. AVs operate across six levels of automation, from Level 0 (manual driving) to Level 5 (fully automated).

Data-Driven Approaches: Early rule-based path planning has evolved into data-driven models like reinforcement learning. Newer architectures integrate sensor data and use deep learning to optimize decision-making.

Knowledge-Driven Models: The integration of LLMs into autonomous driving models is a recent development. These models analyze data, predict vehicle actions, and generate explanations for decision-making processes. Recent research leverages LLMs to align vision-language models with driving scenarios, such as GPT-Driver, DriveGPT4, and LMDrive.

3 Deep Learning

Before we begin to implement any models, it is essential to establish a solid foundational knowledge of deep learning. Deep learning enables computational models, which consist of several processing layers, to acquire data representations across various levels of abstraction(LeCun, Bengio, and Hinton 2015). Understanding these fundamental concepts, such as traditional neural networks, is essential before advancing to its complex applications.

3.1 Traditional Neural Network Architectures

3.1.1 Fully Connected Layer

A fully connected layer (FC), also known as a dense layer, is a fundamental type of neural network layer in which each neuron is linked to all neurons in the preceding layer as shown in Figure 1. This setup ensures that the input features are combined in every possible way to pass information through the network. Its application, such as CNN architecture (Raitoharju 2022), contains one or several fully connected layers, depending on the depth of the architecture (Basha et al. 2020).

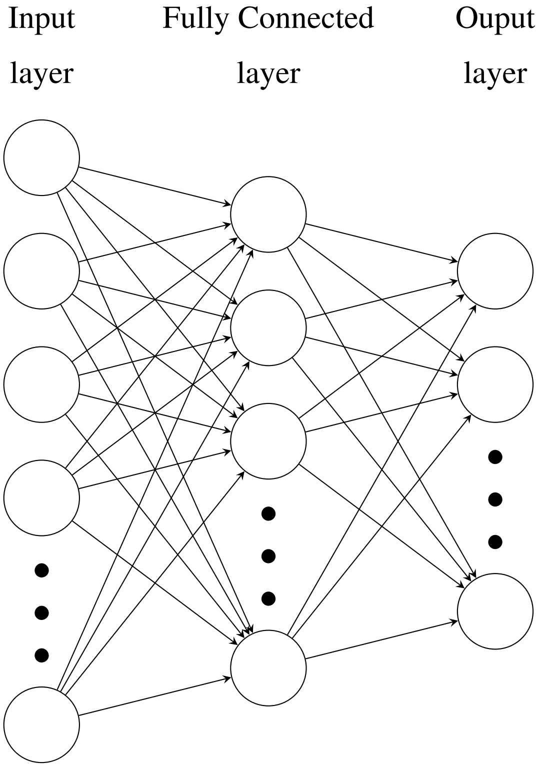


Figure 1: Fully Connected Layer

The mathematical operations performed by such a layer can be described as follows:

$$\mathbf{FC}(\mathbf{x};\mathbf{W}) = \sigma(\mathbf{W}\mathbf{x}+\mathbf{b}) \tag{3.1}$$

Where $\mathbf{Wx}$ represents the matrix multiplication between the weights $\mathbf{W}$ and the input vector $\mathbf{x}$. Moreover, $\mathbf{Wx}+\mathbf{b}$ adds the bias $\mathbf{b}$ to the result of $\mathbf{Wx}$. It represents the input layer to the fully connected layer(also called the hidden layer). Finally, the activate function σ is applied element-wise to the result of $\mathbf{Wx}+\mathbf{b}$ and is responsible for introducing non-linearity into the model, which allows the network to learn and model more complex functions.

3.1.2 Multilayer Perceptron

A multilayer perceptron (MLP) is a type of artificial neural network that is widely used in machine learning for a variety of tasks, such as classification, regression, and feature learning. Most researchers preferred MLP as their neural network topology (Bose 2007). Such a network usually includes fully connected layers and is equipped with nonlinear activation functions. It is structured into at least three layers (input layer, hidden layer, and output layer).

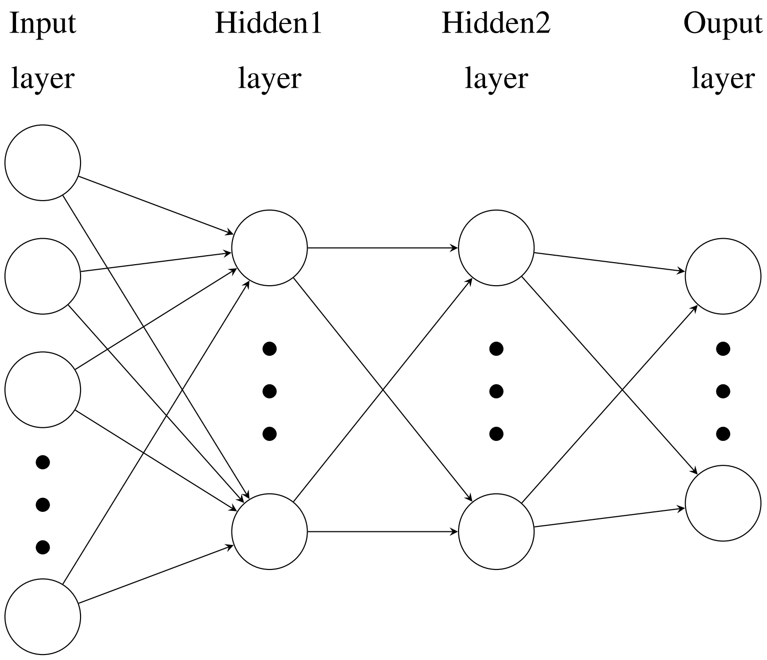


Figure 2: Multilayer Perceptron with 2 fully connected layers

However, if the MLP has only 1 fully connected layer, it is equivalent to a single fully connected layer:

$$\mathbf{MLP}^{(1)}(\mathbf{x};\mathbf{W}) = \mathbf{FC}(\mathbf{x};\mathbf{W}) \tag{3.2}$$

If the MLP has 2 fully connected layers, as shown in Figure 2, the equations should be changed to:

$$\begin{aligned} h^{(1)}(x;W^{(1)}) &= \sigma^{(1)}(\mathbf{W}^{(\mathbf{1})}\mathbf{x} + \mathbf{b}^{(1)}) \\ h^{(2)}(h^{(1)};W^{(2)}) &= \sigma^{(2)}(\mathbf{W}^{(\mathbf{2})}h^{(1)} + \mathbf{b}^{(2)}) \\ \mathbf{MLP}^{(2)}(\mathbf{x};\mathbf{W}) &= h^{(2)}(h^{(1)}(x;W^{(1)});W^{(2)}) \end{aligned}$$

Where $h^{(1)}$ is the output of the first fully connected layer, with $\sigma^{(1)}$ as its activation function, same as $h^{(2)}$ but with $h^{(1)}$ as input. $\mathbf{W}^{(1)}$ is the weight matrices of first layers, while $\mathbf{W}^{(2)}$ is the matrices of second layers. Similarly $\mathbf{b}^{(1)}$ and $\mathbf{b}^{(2)}$ is the bias vectors of the first and second layers.

Note that the notation $\mathbf{MLP}^{(n)}(\mathbf{x};\mathbf{W})$ generally refers to an MLP with n layers, where n indicates the number of fully connected layers in the network.

3.1.3 The GRU Network

Efforts in the simplification of Long Short-Term Memory(LSTM)(Hochreiter and Schmidhuber December 1997a) networks have given rise to the Gated Recurrent Unit (GRU)(Cho et al. 2014) as shown in Figure 3, a model predicated on two multiplicative gates.

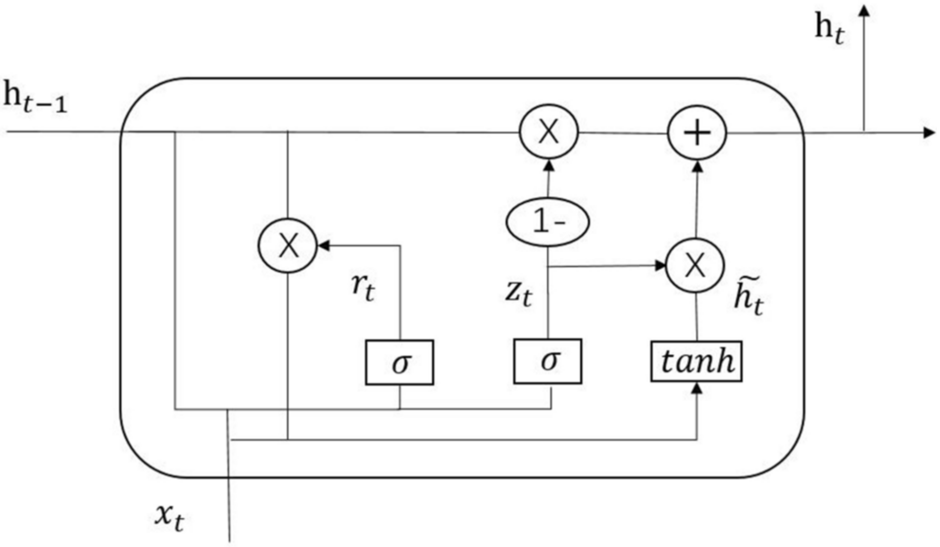


Figure 3: The Architecture of GRU(N. Li et al. May 2021)

Although this model features a reduced gating mechanism, empirical studies suggest that GRUs exhibit performance on par with LSTMs across various machine-learning applications such as GRNN(Chung et al. 2014). The standard GRU Network Equations is defined as (Ravanelli et al. April 2018):

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z), \tag{3.3a}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r), \tag{3.3b}$$

$$\widetilde{h_t} = \tanh(W_h x_t + U_h (h_{t-1} \odot r_t) + b_h), \tag{3.3c}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \widetilde{h_t}. \tag{3.3d}$$

In these expressions, z_t is the update gates, and r_t signifies the reset gates. h_t denotes the state vector at time t. The operator $\odot$ indicates element-wise multiplication. The gates are activated by logistic sigmoid functions σ, limiting update gates and reset gates values between 0 and 1. The candidate state $\widetilde{h_t}$ undergoes transformation via a hyperbolic tangent function.

The network inputs are represented by the vector x_t, such as speech feature vectors, and the parameters encompass feed-forward matrices W_z, W_r, W_h, alongside recurrent matrices U_z, U_r, U_h. Additionally, b_z, b_r, and b_h are bias vectors, integrated before the application of non-linear functions.

As delineated in Eq. 3.3d, h_t is computed as a linear blend of the prior activation h_{t-1} and the candidate state $\widetilde{h_t}$, with proportions determined by the update gate z_t. This mechanism is crucial for encoding long-term dependencies within the network. A high value of z_t maintains the previous state substantially unchanged, facilitating the preservation of information over prolonged periods. Conversely, a lower value of z_t allows for greater influence from the candidate state $\widetilde{h_t}$, reflecting more immediate inputs and recent hidden states. The influence of the reset gate r_t is also notable, enabling the model to potentially discard past information by forgetting earlier computed states. Noteworthily, both the TransFuser (Chitta et al. 2022) and LMDrive (Shao et al. 2023) models employ GRU neural networks to predict waypoints.

3.2 Transformer

The Transformer (Vaswani et al. 2023) is a deep learning model. It has become the foundation for many advanced models in natural language processing (NLP). For example, a language model based on the Transformer architecture BERT, known as Bidirectional Encoder Representations from Transformers (Devlin et al. 2019b), has achieved remarkable performance in NLP. Unlike the recurrent neural network (RNN) suffers from the vanishing or exploding gradient problem(Pascanu, Mikolov, and Bengio 2013), where gradients diminish or grow exponentially over time during training. This limitation led to the development of more advanced architectures like gated recurrent neural networks (GRNNs)(Chung et al. 2014), Long Short-Term Memory (LSTM)(Hochreiter and Schmidhuber November 1997b) which have solidified their status as leading methodologies in sequence modeling and transduction tasks, notably in language modeling and machine translation (Bahdanau, Cho, and Bengio 2016; Cho et al. 2014).

However, GRNNs and LSTM still have their own shortcomings, such as complexity and computational cost, difficulty in capturing global dependencies, and sequential processing.

Transformer resolved many of these issues by relying on attention mechanisms, bypassing the need for recurrence entirely in the architecture. The key innovation in Transformer is the self-attention mechanism (The architecture on the right in Figure 4), which enables the model to assess the significance of different words within the input data, regardless of their order. This breakthrough enabled more efficient training by leveraging parallel computing, thereby drastically reducing training times and improving performance on tasks requiring an understanding of long-range dependencies in text.

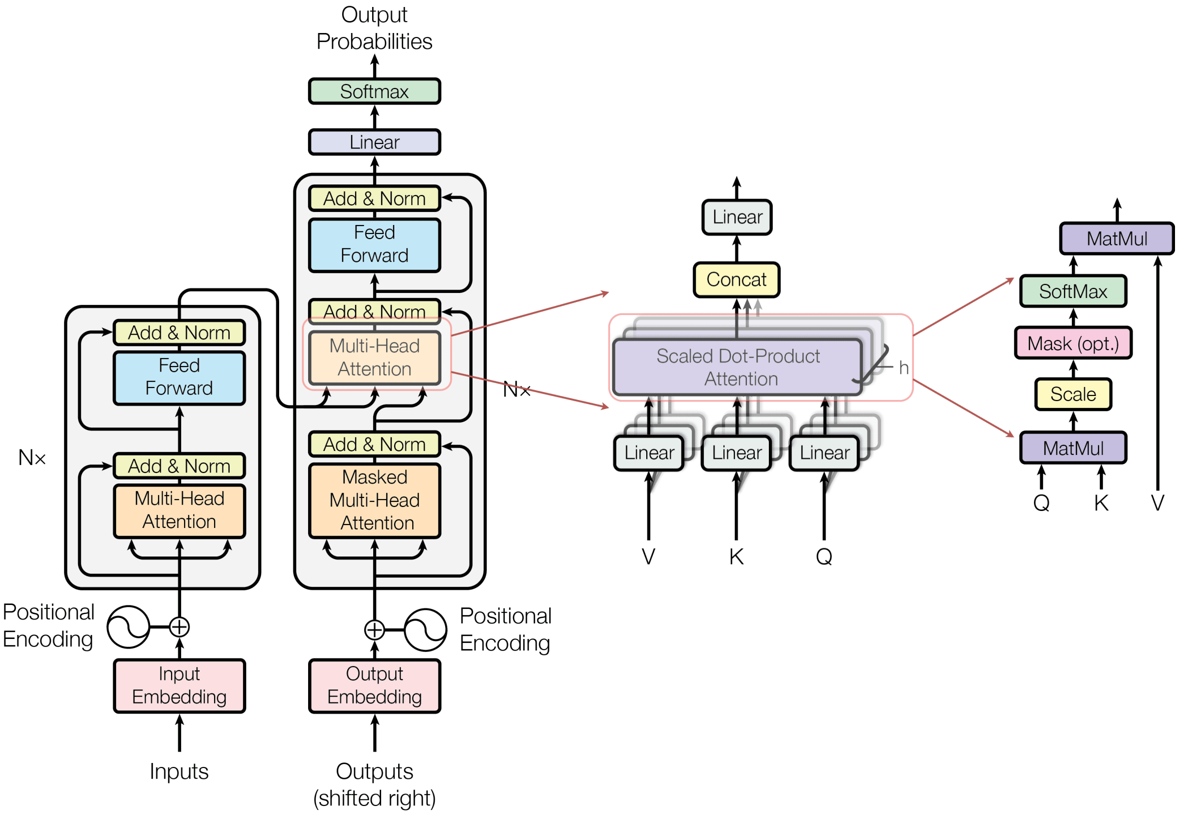


Figure 4: Transformer(Vaswani et al. 2023)

3.2.1 Self-Attention Mechanism

As we mentioned before the self-attention mechanism (also Scaled Dot-Product Attention) is the core of the Transformer architecture. In this section, we try to simplify the mechanism with mathematical equations. The self-attention mechanism takes 3 vectors as input Query (Q), Key(K), and Value(V), which extracted for each word. It outputs a weighted sum of the value vectors, where the weights assigned to each value depend on the compatibility of the query with the corresponding keys.

For example, we have a sentence: "She sells seashells on the seashore.". The query represents the current word or token $Q_i =$ sells in which we are interested. In our sentence, the Key(K) could be "She" "seashells" "on" and "seashore". The value is the information that we want to associate with each key. It provides additional context or meaning to the query. In our sentence, the Value(V) could be the parts of speech, syntactic role, or any other relevant information corresponding to each key. The self-attention can be defined as the following equation:

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{3.4}$$

Where d_k is a dimension of each Query (Q), Key(K), and Value(V) with a dimension of d_v. Firstly, We calculate the dot products of each query with all keys, aiming to measure the similarity between the query's context and the key's corresponding value. This similarity measure is used to determine how much attention or weight should be given to corresponding values. Secondly, we scale these products down by dividing by $\sqrt{d_k}$, which is an important step for stabilizing the training of the model.

However, the reason to use the square root in the denominator ($\sqrt{d_k}$) is to counteract the linear growth of the variance without overly diminishing the magnitude of the dot products. By scaling with $\sqrt{d_k}$, the variance of the resulting scaled dot products is normalized to a constant (not growing with d_k). This keeps the values fed into the softmax function in a range that prevents numerical issues like underflow or overflow. It is a common method used when dealing with very small or large exponents in the softmax. And lastly, we use an activation function(softmax) to determine the weights assigned to the values.

3.2.2 Multi-Head Attention

A single attention mechanism computes attention weights based on the similarity between a query and all keys in the input sequence. However, in multi-head attention, this process is repeated multiple times, each with its own set of learned parameters.

$$\text{MultiHead} = \textbf{Concat}(head_1, ...head_h) * W^O \tag{3.5}$$

$$\text{Where head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{3.6}$$

3.2.3 Transformer Encoder and Decoder

In the Transformer model, the term feedforward neural network(FNN) is used to describe a specific component of the architecture that operates at each position identically and independently. This component consists of two linear (fully connected layers 3.1) transformations separated by a nonlinear activation function, typically ReLU or GELU (Sharma, Sharma, and Athaiya 2017). This closely resembles the architecture of an MLP Eq.3.2, where the FNN effectively is an MLP applied independently to each position. The "Add & Norm" structure with a feed-forward network resembles a two-layer MLP with skip connections and layer normalization.

$$\mathrm{y} = \text{LayerNorm}(\text{MultiHead}(Q, K, V) + \mathrm{x}) \tag{3.7a}$$

$$\text{FNN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{3.7b}$$

$$\textbf{TE}(x) = \text{LayerNorm}(\text{FNN}(\mathrm{y}) + \mathrm{y}) \tag{3.7c}$$

Similar to Transformer encoder (TE), Transformer decoder (TD) has one extra Masked Multi-Head Attention written as:

$$\text{Attention}_M(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + M\right)V$$

$$\text{MultiHead}_M = \textbf{Concat}(head_1, ...head_h) * W^O$$

$$\text{head}_i = \text{Attention}_M(QW_i^Q, KW_i^K, VW_i^V)$$

where M is the mask matrix, ensuring that the softmax function zeros out any inappropriate. here are the equations of TD.

$$\mathrm{y}_M = \text{LayerNorm}(\text{MultiHead}_M(\mathrm{Q}_{\text{dec}}, \mathrm{K}_{\text{dec}}, \mathrm{V}_{\text{dec}}) + \mathrm{x}) \tag{3.8a}$$

$$\mathrm{y} = \text{LayerNorm}(\text{MultiHead}(\mathrm{Q}_{\mathrm{y}_M}, \mathrm{K}_{\text{enc}}, \mathrm{V}_{\text{enc}}) + \mathrm{y}_M) \tag{3.8b}$$

$$\text{FNN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{3.8c}$$

$$\textbf{TD}(x) = \text{LayerNorm}(\text{FNN}(\mathrm{y}) + \mathrm{y}) \tag{3.8d}$$

3.3 Vision Transformer

Vision Transformer (ViT)(Dosovitskiy et al. 2021) applies the Transformer (Vaswani et al. 2023) architecture to the field of computer vision, where convolutional architectures maintain their dominance (He et al. 2015). Motivated by the achievements in natural language processing (NLP), some studies try to integrate convolutional neural network (CNN)-based architectures with self-attention mechanisms (X. Wang et al. 2018; Carion et al. 2020). Some try to remove convolutional neural networks completely (H. Wang et al. 2020). But They didn't outperform the residual neural network(ResNet) in large-scale image recognition (Q. Xie et al. 2020; Kolesnikov et al. 2020).

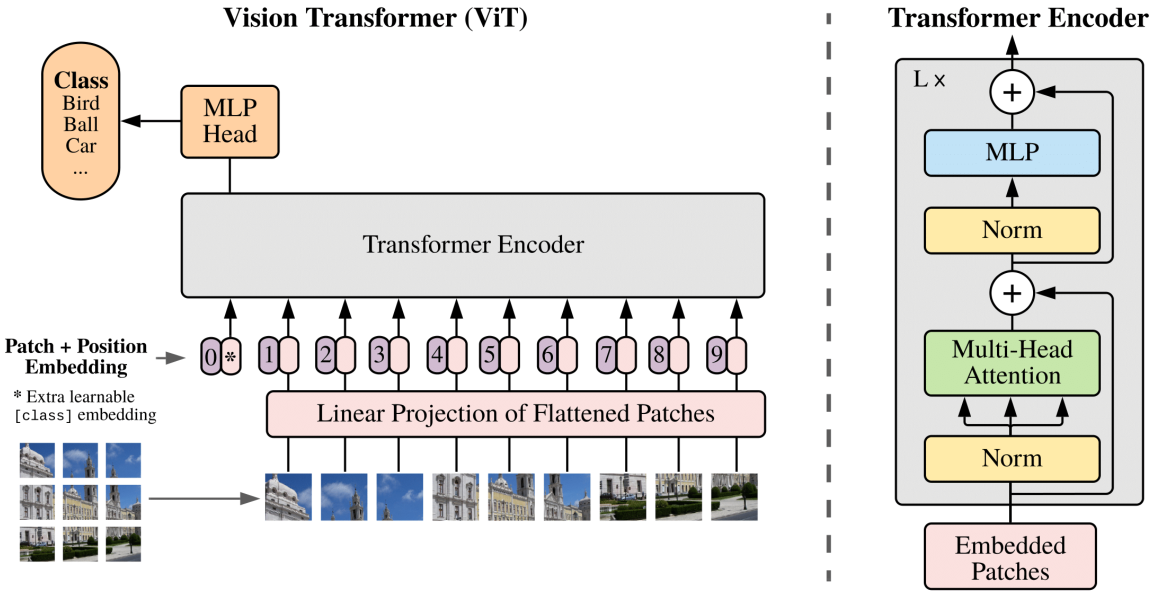


Figure 5: ViT(Dosovitskiy et al. 2021)

Unlike traditional CNNs that process images in parts, ViTs treat images as sequences of patches and apply self-attention mechanisms across these patches, offering an alternative that can achieve competitive results in image classification tasks.

3.3.1 Input Embedding

Given an input image I of size $H \times W \times C$ (height, width, and channels), ViT first divides the image into a grid of fixed-size patches. Let P denote the number of patches, and $P = \frac{HW}{\text{patch\_size}^2}$. Each patch is then linearly projected into a lower-dimensional embedding space

d_{patch} using a learnable weight matrix W_{patch}:

$$\text{Patch}_{ij} = \text{flatten}(I[\text{pos}(i,j)]) \cdot W_{\text{patch}},$$

where $\text{pos}(i,j)$ retrieves the spatial coordinates of the patch at position (i,j), and $\text{flatten}(\cdot)$ flattens the patch into a 1D vector.

3.3.2 Positional Encoding

Since Transformer don't inherently capture spatial information, positional encodings are added to the patch embeddings to provide information about their positions. The positional encoding *PE* is typically calculated as a combination of sine and cosine functions of different frequencies:

$$PE_{(p,2i)} = \sin(p/10000^{2i/d_{\text{patch}}}),$$
$$PE_{(p,2i+1)} = \cos(p/10000^{2i/d_{\text{patch}}}),$$

where p is the position and i is the dimension.

3.3.3 Transformer Encoder

The patch embeddings with positional encodings are then fed into a Transformer encoder. The encoder consists of multiple layers, each composed of a multi-head self-attention mechanism followed by a feedforward neural network (FNN). The output of each layer is computed as follows:

$$y = \text{LayerNorm}_1(\text{MultiHead}(\text{LayerNorm}_2(\text{Patch} + PE)))$$
$$\text{LayerNorm}_3(\text{FNN}(\text{LayerNorm}_4(y))) + \text{Patch},$$

where LayerNorm denotes layer normalization, and y is the first layer.

3.3.4 Classification Head

Finally, the representation of the special '[CLS]' token, typically from the last layer of the encoder, is extracted and passed through a linear layer followed by a softmax activation function to predict the class probabilities:

$$\mathrm{softmax}(\mathrm{Linear}(\mathrm{CLS}))$$

3.4 Summary

We provide a comprehensive analysis of deep learning models and their practical uses, offering foundational insights into architectures like Transformer and Vision Transformer. And also we explore key concepts and foundational models in deep learning.

Fully Connected Layer means every neuron is connected to each neuron in the previous layer, combining all input features and passing information through the network. This allows complex functions to be learned via matrix multiplication, bias addition, and activation functions. **Multilayer Perceptron (MLP)** extends the fully connected layer concept with multiple layers, each adding complexity by incorporating activation functions. **GRU Network** is a simplified alternative to LSTM networks, utilizing multiplicative gates (update and reset gates) to facilitate long-term dependencies and discard outdated information

Transformer introduces an attention mechanism that allows processing sequences without recurrence, overcoming the limitations of previous architectures. The self-attention mechanism compares word similarity to generate weighted sums, scaling outputs for stability.

Vision Transformer adapts the transformer architecture for computer vision by treating images as patches instead of sequences, enabling competitive performance in image classification tasks.

4 Deep Learning Models in Autonomous Driving

Deep learning plays a significant role in the current autonomous driving revolution both in academia and industry (Grigorescu et al. 2020). As we mentioned, many models that integrate deep learning approaches with autonomous driving in Chapter 2, we elaborate on the selected models TransFuser and LMDrive in more detail. Both models are designed for motion planning tasks.

4.1 TransFuser

TransFuser (Chitta et al. 2022) is an advanced architecture designed for autonomous driving, utilizing a novel approach that integrates multiple sensor modalities, specifically images, and LiDAR, using Transformer technology. This integration enables the model to effectively navigate complex environments by fusing rich, contextual information from both sensor types. Overall, the TransFuser architecture shown in Figure 6 can be represented simply as equation 4.3.

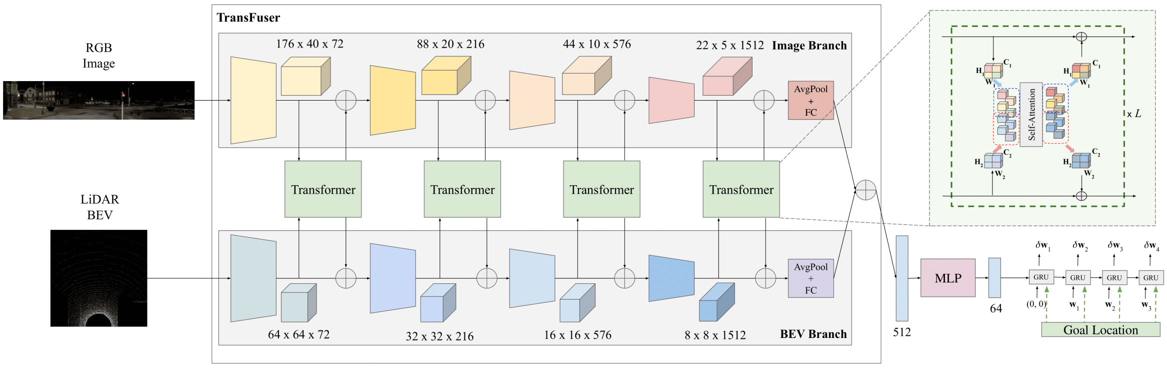


Figure 6: Architecture of TransFuser(Chitta et al. 2022)

$$\mathbf{w}_t = \mathbf{TransFuser}(\mathbf{I}, \mathbf{L}; \mathbf{W}) \tag{4.1}$$

Where $\mathbf{w}_t$ is the predicted waypoint containing a sequence of 2D coordinators $\{x_t, y_t\ \}$. The $(\mathbf{I}, \mathbf{L})$ are RGB camera images and LiDAR cloud points, respectively. $\mathbf{W}$ is the weight of the model. The TransFuser architecture comprises two primary components Multi-Modal Fusion Transformer and Auto-Regressive waypoint Prediction Network.

4.1.1 Multi-Modal Fusion Transformer

Similar to Vision Transformer(in Section 3.3), the Multi-Modal Fusion Transformer(MMFT) uses Transformer (Vaswani et al. 2023) to extract features from the image and LiDAR data streams. It processes these features at multiple scales within the network, ensuring that the fusion captures both high-level and low-level details relevant to driving decisions.

$$\begin{gathered}F_t = \mathbf{ResNet}(\mathbf{X}) \\ Q,K,V = F_tM^q, F_tM^k, F_tM^v \\ A_t = \mathbf{MultiHead}(Q,K,V) \\ F_{t+1} = \mathbf{MLP}(A_t) + F_t\end{gathered}$$

Where $F_t \in \mathbb{R}^{N\times D_f}$ is the input sequence and here N denotes the total number of tokens within the sequence, with each token being characterized by a feature vector of dimensionality D_f. In this context, X will be image data or LiDAR data and apply the ResNet (He et al. 2015) operation. The Query (Q), Key(K), and Value(V) apply linear projections to F_t and weight matrices ($M^q \in \mathbb{R}^{D_f\times D_q}$, $M^k \in \mathbb{R}^{D_f\times D_k}$ and $M^v \in \mathbb{R}^{D_f\times D_v}$). Then, we get the output features F_{t+1} by using Multi-Head Attention Eq.3.5 and Multilayer Perception Eq.3.2. Finally, we model the **MMFT** by applying fully connected layer Eq.3.1 and average pooling layer(AvgPool).

$$\mathbf{MMFT}(\mathbf{X}) = \mathbf{FC}(\mathbf{AvgPool}(F_{t+1}) \tag{4.2}$$

4.1.2 Auto-Regressive Waypoint Prediction Network

As depicted in Figure 6, the 512-dimensional feature vector undergoes dimensionality reduction to 64 via a Multilayer Perceptron (MLP) with two hidden layers of 256 and 128 units each, aimed at enhancing computational efficiency. Subsequently, the transformed vector is fed into the auto-regressive waypoint network, which is realized using GRUs (Cho et al. 2014). Lastly, we complete the TransFuser model as Eq.4.3.

$$\mathbf{TransFuser}(\mathbf{I},\mathbf{L}) = \mathbf{GRU}(\mathbf{MLP}(\mathbf{MMFT}(\mathbf{I}) + \mathbf{MMFT}(\mathbf{L})) \tag{4.3}$$

4.2 LMDrive

LMDrive(Shao et al. 2023) is a pioneering framework designed to interpret and execute high-level driving instructions using LLM. This innovative model consists of two primary components which are integrated to process sensory data and linguistic commands for vehicle control.

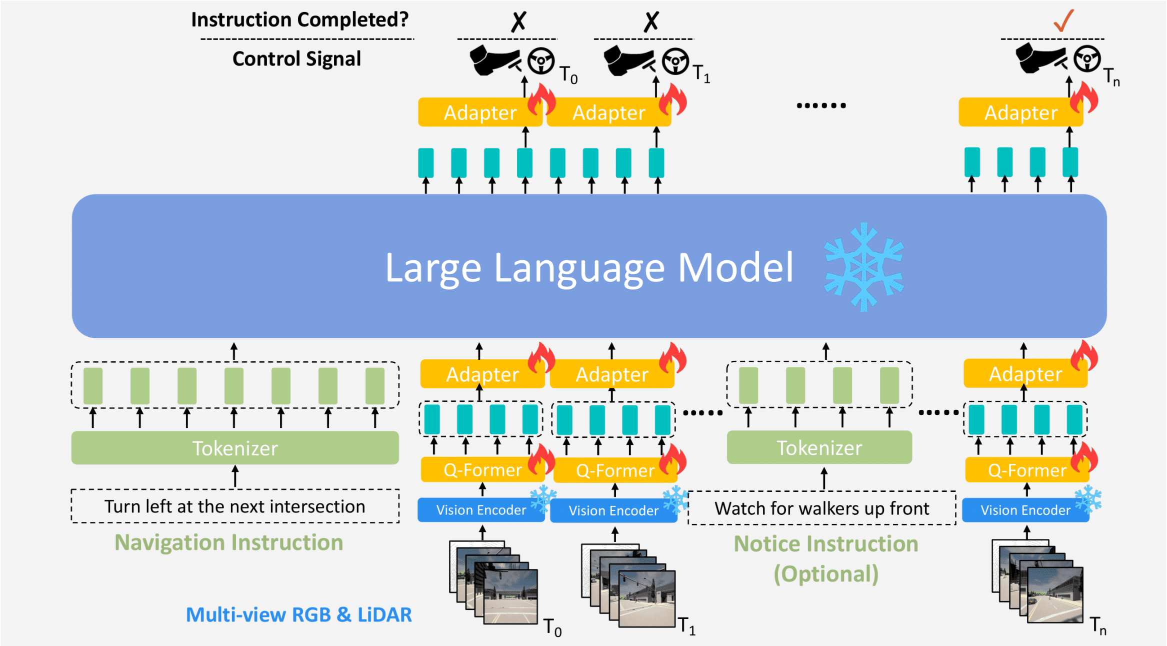


Figure 7: Architecture of LMDrive(Shao et al. 2023)

similar to TransFuser, LMDrive takes RGB images(I), LiDAR (L) and natural language navigation instructions(S) as inputs, and outputs waypoints(w_t) as control signals for driving actions in real-time.

$$w_t = \mathbf{LMDrive}(I, L, S; W) \tag{4.4}$$

4.2.1 Vision Encoder

Similar to the Multi-Modal Fusion Transformer of TransFuser, LMDrive uses the vision encoder for processing the visual and spatial data obtained from RGB image cameras and LiDAR.

Vision Encoder(VE) takes RGB images(I) and LiDAR (L) as inputs and outputs visual tokens (V_{tokens}) which include BEV tokens, waypoint tokens, and traffic light tokens. VE can be defined as:

$$\mathbf{VE}(I, L) = V_{tokens} \tag{4.5}$$

Where the image date $I \in \mathbb{R}^{3\times H\times W}$ are applied CNN backbone ResNet(He et al. 2016), and it produces a feature map $f \in \mathbb{R}^{d\times H\times W}$ with lower-resolution. For the LiDAR input L, the 3D backbone PointPillars(Lang et al. 2019) are used to process the raw point cloud data into ego-centered LiDAR features, where each pillar contains points within a 0.25m × 0.25m area. PointNet(Qi et al. 2018) is then employed to aggregate these features and downsample the feature map to dimensions $C \times H \times W$, which are then utilized as BEV queries.

$$\begin{gathered} Q = \text{PointNet}(\text{PointPillars}(L)) \\ V, K = \mathbf{TE}(\text{Flatten}(\text{ResNet}(I))) \\ V_{tokens} = \mathbf{TD}(Q, K, V) \end{gathered}$$

4.2.2 Large Language Model

After processing the visual data, LMDrive utilizes a sophisticated large language model (LLM) to understand and execute driving instructions provided in natural language. This integration allows the system to interpret instructions contextually and make informed driving decisions.

Large Language model (**LLM**) takes language instructions sentence (S) and vision tokens V_{tokens} as inputs, and outputs waypoints((w_t)) as control signals for driving actions in real-time. LLM in LMDrive can be defined as:

$$\begin{gathered} T_{tokens} = \mathbf{Tokenizer}(S) \\ V'_{tokens} = \mathbf{MLP}^{(2)}(\mathbf{QFormer}(V_{tokens})) \end{gathered}$$

$$\text{LLM}(S, V_{tokens}) = T_{tokens} + V'_{tokens} \tag{4.6}$$

Finally the LMDrive:

$$\mathbf{LMDrive}(I, L, S) = \mathbf{MLP}^{(2)}(LLM(S, \mathbf{VE}(I, L))) \tag{4.7}$$

4.3 Summary

We provide an overview of two models used in autonomous driving, TransFuser and LM-Drive. These two models offer different but complementary ways to enhance motion planning for autonomous vehicles **TransFuser** integrates images and LiDAR data through a Transformer architecture for accurate navigation. It consists of two main components: the Multi-Modal Fusion Transformer (MMFT) and the Auto-Regressive Waypoint Prediction Network. **LMDrive** is a framework that combines sensory data and natural language instructions for driving control. the inputs include RGB images, LiDAR data, and high-level language instructions. It has a vision encoder that processes RGB and LiDAR data to generate tokens for use in decision-making. Also, It employs a large language model (LLM) to interpret navigation instructions and integrates it with sensory data to generate control signals in real time

5 Methodology

We use the core methodology of simulation (Paul and Balmer 1993) to address our research questions 2, 3, and 4. The main goal of simulation modeling is to assist the final decision-maker in resolving an issue (Shannon 1998). For example, we use CARLA (Dosovitskiy et al. 2017) to help us solve evaluation problems that are too perilous for conducting real-world experiments. Consequently, the method has gained recognition as an effective tool for problem-solving within the research community, with widespread agreement on its adoption for modeling purposes (Eldabi, Irani, and Paul February 2002). A typical process of simulation is illustrated in Figure 8.

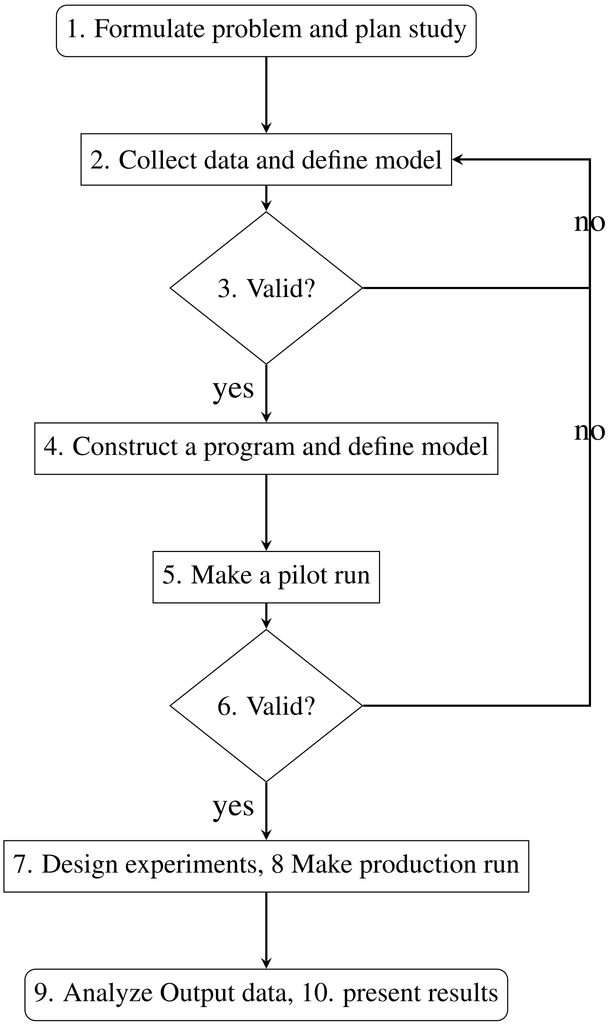

1. Formulate problem and plan study
2. Collect data and define model
no
3. Valid?
yes
4. Construct a program and define model
no
5. Make a pilot run
6. Valid?
yes
7. Design experiments, 8 Make production run
9. Analyze Output data, 10. present results

Figure 8: A typical simulation process(Law and Kelton 1991)

Importantly, a methodology should not dominate the research process under any circumstances confirmed by Quinn, Mintzberg, and James 1988. Therefore, we categorize the simulation process into 3 steps. 1. formulate problems. 2. implement the selected model and simulate the program as elaborated in Chapter 6. 3. evaluate the results in Chapter 7.

The first step of the simulation process (Law and Kelton 1991) is to formulate the problems by establishing our objectives and identifying the particular matters that warrant consideration.

Research Question 1: Let A be the set of all approaches, M the set of large language models applicable to autonomous driving, and S the simulation environments. We define a function $f : A \times M \times S \to \mathbb{R}$ that measures the suitability of each combination of approaches, models, and simulators for autonomous driving tasks. The objective is to find:

$$\max_{a \in A, m \in M, s \in S} f(a,m,s) \tag{5.1}$$

So far we have conducted a systematic literature review on Chapter 2. We explored different approaches(A) and open source simulators(S) and custom simulators with language models(M).

First of all, we decided CARLA should be our simulator($s = \text{Carla}$) for 3 compelling reasons. 1. CARLA offers 3D maps and dynamic weather systems that are more realistic than the other simulators. 2. CARLA includes a wide range of sensor types, such as LIDAR, radar, cameras, and GPS, which accurately replicate the inputs an autonomous vehicle would use. 3. CARLA has strong community support and frequent updates.

Secondly, we have strong justifications for taking LMDrive as our focusing approach($a = \text{LMDrive}$). We run experiments with a prompting approach from GPT-driver(Mao, Qian, et al. 2023) (see in Appendix8) on the Carla simulator, which fails steps 5 and 6(see in Figure8). The hindrance of prompting approaches run on the Carla simulator due to the reason that they are designed to 2 dimension simulators such as HighEnv(Leurent 2018), and the prompts need to describe very specific details about the environment. Moreover, there is a notable lack of a comprehensive benchmark consisting of substantial linguistic prompts(Chen, Wu, et al. 2023). Ultimately, the LMDrive(Shao et al. 2023) model was chosen for further evaluation due to its open-source nature, allowing a thorough understanding of its architecture. It also has credible explanations in their experiments that LLaVA-v1.5(Liu, Li, Li, et al. 2023) outperforms the other LLM models. Therefore $m = \text{LLaVA1.5}$.

Research Question 2: Given a simulation environment ($s = \text{Carla}$), approach($a = \text{LMDrive}$), and large language model ($m = \text{LLaVA1.5}$). Let $g : X \to T$ be a development function within s, a, m to its performance on T where T is a set of motion planning tasks and X is necessary sensor data such as RBG image and LiDAR cloud points. Our gold can be formulated as:

$$\mathrm{g}(X; s, a, m) \tag{5.2}$$

The full implementation of our goal function is described in Section 6.2.

Research Question 3: Let $a = \text{LMDrive}$ and $b = \text{TransFuser}$ represent the knowledge-driven and data-driven model respectively. Define $p(x;\theta)$ as a performance output that evaluates a set of metrics(θ defined in Section 6.3). The objective is to evaluate:

$$\text{Compare}(\{p(k;\theta)\}, \{p(b;\theta)\}) \tag{5.3}$$

This formulation aims to quantify and compare the effectiveness and efficiency across model types. The detailed comparison is shown in Section 7.1.

Research Question 4: For a set of predefined routes (R shown in Section 6.1.1) where decisions are made by a large language model $m = \text{LLaVA1.5}$ from the knowledge-driven approaches $a = \text{LMDrive}$, define the interpretability function $E : a \times R \to \{0, 1\}$, where the output represents whether the interpretability successed(1) or fail(0) to describe a scenario. The objective is to asses:

$$\text{Assess}(E) \tag{5.4}$$

The analysis of the interpretability results is shown in Section 7.2.

6 Experiments

In the chapter, we answer the research question 2, formulated in Eq.(5.2). We elaborate on the simulator and scenarios we used for the experiments and then demonstrate the implementation of deep learning models integrated into the CARLA simulation environment. Lastly, we define our metric to evaluate the models.

6.1 CARLA Simulation

For the experiments, we utilize version 0.9.10.1 of the open-source CARLA simulator (Dosovitskiy et al. 2017). It is famous for its realistic driving environments and flexible testing capabilities. CARLA provides a rich set of built-in maps, such as the 3D model of Town01, derived from an OpenDRIVE file (ASAM e.V. 2023). This map provides road layouts, lanes, and junctions, etc., which are ideal for complex simulation scenarios.

The core component of CARLA is to manage basic vehicle dynamics, sensor simulation, and environmental conditions such as weather. But it is not enough, we need extra 2 components 1. Carla Leaderboard (Leaderboard 2024) and 2. ScenarioRunner (ScenarioRunner 2024). The first component is a platform designed to benchmark the performance of autonomous driving systems in a simulated environment. It utilizes the scenarios created and managed by the ScenarioRunner to conduct standardized tests. It automates the process of running autonomous agents through various scenarios to evaluate their performance based on specified metrics. The second component is a tool used within the CARLA simulator to define, execute, and manage different traffic scenarios.

Inspired by the Longest6 Benchmark (Chitta et al. 2022), we have simplified this benchmark to suit our specific research interests, such as the time-consuming of the models. This adjustment allows for a more focused evaluation of critical aspects of autonomous driving relevant to our study without the redundant time complexity for the test environments.

6.1.1 Scenarios

The main evaluation focuses on navigating a vehicle on predefined routes in the Town01 environment, with 3 scenarios(or routes) designed to test the ability of each model to handle different driving situations. These scenarios include navigating through traffic jams, making left and right turns, responding to vehicles running red lights, and obstacle avoidance. The routes, marked by sparse GPS coordinates provided by a global planner, challenge the agents to adhere to traffic laws while managing interactions with numerous dynamic agents within a set timeframe.

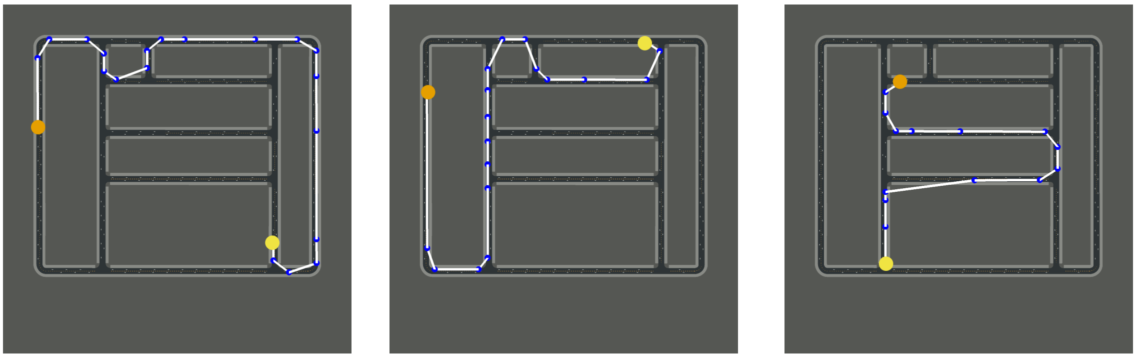
Each route starts from a specific position (marked in yellow) and follows a designated path (white line) to a goal location (orange point), as exemplified in Figure 9. This setup not only tests the navigational accuracy and decision-making capabilities of models but also their ability to integrate sensor inputs with dynamic traffic elements in a realistic virtual setting.

Each route varies in the number of traffic lights and turns. These elements are critical in testing how well the model handles decision-making at intersections and during lane changes or turns.

Route01 Fig.(9a): With 6 traffic lights and 8 turns over a distance of 1130.24 meters, this route is designed to test the capability of the model to handle a moderate number of decision points (traffic lights and turns) over a longer distance.

Route02 Fig.(9b): This route has 9 traffic lights but the same number of turns as Route 01 packed into a slightly shorter distance (1014.57 meters). The increased number of traffic lights could test the response of the model to frequent stop-and-go scenarios and prioritization at intersections.

Route03 Fig.(9c): The shortest route at 731.50 meters with 5 traffic lights and only 5 turns. This setup is intended to evaluate the model's efficiency and decision-making in a less complex but more streamlined path, focusing on the efficiency of navigation with fewer interruptions.



(a) Route 1 (b) Route 2 (c) Route 3

Figure 9: Ego-vehicle start at yellow point, end at orange point follow pre-designed routes along with white line

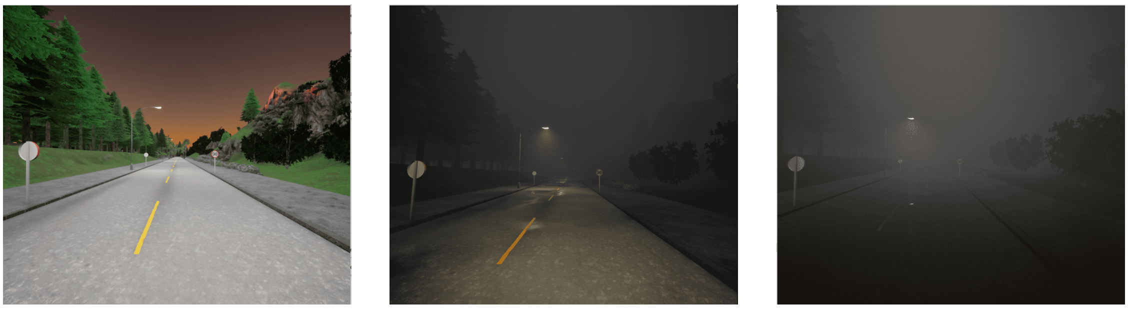
6.1.2 Conflict Scenarios

In addition to testing the performance of autonomous driving agents in conflict scenarios, we exploit the weather system that the simulator provided. Gold et al. June 2017 proposes a Classification for level-3 conflict scenarios based on a scale of 1 to 3 of 4 factors such as Urgency, Predictability, Criticality, and Driver response. However, we are interested in the conflicts that are shown in the table 3.

No.	Name	Urgency	Predictability	Criticality	Driver Response
2	Sensor failure	3	1	2-3	1-2
9	Danger zone/obstacle ahead(detected by on-board sensors)	3	1	1-3	1-3
10	Loss of reference signals (e.g. lane markings missing)	3	1	2-3	1

Table 3: Classification of different testing scenarios (Gold et al. June 2017)

In order to test the performances of the models in such conflicts, we use the weather system indirectly to represent the conflicts. For example, our extreme condition is shown in Figure 10c failed to see the landmark of the speed limit compared to other conditions. The Town01 has already included the Danger zone such as a bumpy route condition. The configuration also contains heavy rain, which leads to the LiDAR sensor's fail rate being relatively high compared to clear weather(Dreissig et al. 2023).



(a) Default Weather (b) Medium Weather (c) Extreme Weather

Figure 10: The view of 3 conditions of weather

As Table 4 shows the configuration of different weather corresponding to Figure 10, which compares different weather conditions in the same place.

Conflicts	Config	Description
default	cloudiness=0	no cloud
	precipitation=0	no rain
	precipitation deposits=0	no puddles on the road
	sun altitude angle=0	dawn
	fog density=0	no fog
	wetness=0	no humidity
medium	cloudiness=50	medium level overcast by cloud
	precipitation=50	medium rain
	precipitation deposits=50	some puddles on the road
	sun altitude angle=-45	evening
	fog density=50	medium concentration of the fog
	wetness=50	medium humidity percentages of the road
extreme	cloudiness=100	complete overcast
	precipitation=100	heavy rain
	precipitation deposits=100	roads completely capped by rain
	sun altitude angle=-90	midnight
	fog density=100	extreme thickness of the fog
	wetness=100	extreme humidity percentages of the road

Table 4: Configurations of Conflict Scenarios

6.2 Autonomous Agents

Autonomous agent in the CARLA simulator are essentially software entities that represent self-driving cars. It is a blueprint from which all the customized agents should inherit. An autonomous agent instance is loaded into a scenario created by the ScenarioRunner(ScenarioRunner 2024).

TransFuser Agent: We use an pre-train TransFuser (Chitta et al. 2022) as data-driven model. The pre-train weight can be defined as W^T. The training dataset (210GB) includes

3500 training routes in CARLA from Town 01 to 07 and 10HD. The TransFuser model is defined as:

$$\text{TransFuser}(x; W^T)$$

LMDrive Agent: In this research, we employ the open-source LLM LLaVA1.5(Liu 2023). The training dataset consists of several distinct components: First, 558,000 image-text pairs that have been curated from the LAION (Schuhmann et al. 2022), SBU datasets (Ordonez, Kulkarni, and Berg 2011), and Creative Commons (CC)(Commons 2022) with annotations provided by the BLIP(Techie and Designer 2022) captioning system. Additionally, the model utilizes 158,000 pieces of multimodal data, specifically generated through GPT (Developer and Researcher 2023) to simulate instruction-following tasks. The dataset also integrates 450,000 entries derived from a specialized set of visual question-answering (VQA) tasks designed for academic purposes (Scientist and Analyst 2017), alongside 40,000 instances from the ShareGPT dataset (Innovator and Thinker 2024).

Our methodology incorporates the use of the pre-trained model LMDrive-1.0 (OpenDILab Community 2023), specifically developed for dynamic driving environments. This model was trained on a dataset comprising 64,000 data clips, each meticulously collected within the CARLA simulation environment. The dataset for LMDrive-1.0 is richly featured, with each clip containing a single primary navigation instruction supplemented by a dozen ancillary notice instructions. Additionally, the clips encompass a comprehensive sequence of multi-view sensor data from multiple modalities, alongside the requisite vehicle control signals. The temporal scope of these clips varies, ranging from a brief 2 seconds to a more extended period of 20 seconds. For formal definition within the scope of this study, the pre-trained weights of the model are denoted as W^L, and the LMDrive model is defined as:

$$\text{LMDrive}(x; W^L, m)$$

As Table 5 shows, LMDrive requires more advanced hardware, such as an NVIDIA RTX V100 GPU with 32GB VRAM in our setup. This is because it integrates an LLM with 7B parameters, which need at least 16GB VRAM. This comparison highlights the varying computational demands of each model, which directly influence their complex implementation and deployment scalability.

Models	Components	Require
TransFuser	GPU	NVIDIA GeForce RTX 3070 ti laptop with 8 GB
	image branch	ResNet34(He et al. 2016)
	BEV branch	ResNet18(He et al. 2016)
LMDrive	GPU	NVIDIA V100 with 32 GB
	image branch	ResNet-50(He et al. 2016)
	BEV branch	PollarPoint(Lang et al. 2019),PointNet(Qi et al. 2018)
	LLM	llava-v1.5-7b(Liu, Li, Li, et al. 2023)

Table 5: Configurations of TransFuser and LMDrive

6.3 Metrics

Many state-of-the-art models use default metrics from the Carla benchmark (Leaderboard 2024). However, we use route completion from it and evaluate other metrics that we are interested in, such as the number of collisions and the inference time of the model. We define our metrics θ as

$$\theta = \{\mathrm{RC}, \mathrm{NOC}, \mathrm{ST}, \mathrm{GT}, \mathrm{IT}\} \tag{6.1}$$

Where **Route Completion (RC)**, measures the average percentage of the route distance, R_i, that an agent completes across 3 routes. "If an agent strays from the designated lanes for any segment of the route, the RC is reduced by a 1% multiplier based on the distance off route. **Number of Collisions (NOC)**, measures the number of times that collisions occur to the ego vehicle. **System Time (ST)**, measures the time in seconds of a completed scenario that the ego vehicle takes in the real world. **Game Time (GT)**, measures the time in the seconds of the completed scenario that the ego vehicle takes in the simulator world. **Inference Time (IT)**, The ST and GT are indirect methods to evaluate the time that the model takes, which implies how effective the model is.

$$\mathbf{IT} = \frac{GT}{ST}$$

7 Results

This chapter presents the findings from all the evaluation tasks conducted. First of all, we compare the performance of a data-driven model to a knowledge-driven model. Secondly, we evaluate the efficacy of interpretability of a knowledge-driven model.

7.1 Comparison of TransFuser and LMDrive

The research question 3 will be answered in this section by comparing the performance of two autonomous driving models TransFuser (Chitta et al. 2022) and LMDrive (Shao et al. 2023). This is across several metrics defined in Eq.6.1.

7.1.1 No Conflicts Scenarios

We conducted three experiments using both the TransFuser and LMDrive models across the three routes defined in Section 6.1.1 under default configuration conditions. The results indicate that TransFuser outperforms LMDrive across all metrics. Table 6 shows that TransFuser requires an average of 61.01 minutes in system time and demonstrates remarkable performance in both RC and NOC. However, compared to the rule-based agent, the model still faces limitations in inference time.

Routes	Eval	RC	NOC	ST	GT	IT
1	1	100	1	4022.02	735.85	0.182
	2	100	0	4232.86	746.0	0.176
	3	100	1	3833.23	699.8	0.182
	avg.	100	0.66	4029.37	727.21	0.180
2	1	100	2	3986.6	739.45	0.185
	2	95.79	0	4664.34	860.05	0.184
	3	100	0	3364.07	614.7	0.182
	avg.	98.59	0.66	4005	738.06	0.184
3	1	87	0	3612.63	623.06	0.172
	2	100	0	2547.37	437.85	0.171
	3	100	1	2686.65	473.75	0.176
	avg.	95.66	0.33	2948.88	511.55	0.173

Table 6: Results of TransFuser on 3 routes with 3 times evaluation

Conversely, the results for LMDrive are presented in Table 7. LMDrive exhibits a significantly lower completion rate, averaging 16.59%, while TransFuser achieved nearly a 100% completion rate across multiple evaluations (Table 6). Due to the low route completion rate, LMDrive required 43.57 minutes less time than TransFuser. This difference in performance is reflected in their respective inference times(IT). For instance, TransFuser averaged 0.18, indicating efficient processing relative to simulation time, compared to 0.068 of LMDrive, which points to lower runtime efficiency.

Routes	Eval	RC	NOC	ST	GT	IT
1	1	38	2	2290.53	154.95	0.067
	2	5.05	1	377.49	39.85	0.105
	3	16.75	2	5230.78	371.1	0.070
	avg.	19.933	1.66	2632.93	188.63	0.071
2	1	27.27	1	2044.3	135.85	0.066
	2	3.2	1	705.75	44.9	0.063
	3	4.78	3	3448.73	213.35	0.0618
	avg.	11.75	1.66	2066.26	131.36	0.0635
3	1	19.46	1	3356.01	232	0.0691
	2	15.64	0	1839.4	116.05	0.0630
	3	19.2	1	4238.54	292.85	0.0690
	avg.	18.1	0.66	3144.65	213.63	0.0679

Table 7: Results of LMDrive on 3 routes with 3 times evaluation

Finally, we conducted the same experiment using a rule-based model as a baseline to assess how deep learning models impact inference time, as shown in Table 8. The efficiency of the rule-based agent in inference time is attributed to its use of fewer computational resources than deep learning models.

In comparison, the TransFuser model not only demonstrates faster processing times but also greater consistency across various testing scenarios compared to LMDrive. The longer inference times of LMDrive suggest potential inefficiencies in its architecture, possibly due to the complex decision-making processes involving large language models (LLMs) or the less optimized Vision Transformer (ViT). Manual observations revealed that LMDrive frequently repeated errors, such as collisions with non-player characters (NPCs) across different lanes. This behavior indicates that the model is overfitting, being too trained to specific scenarios and thus failing to generalize across diverse situations.

Models	RC	NOC	ST	GT	IT
Rule-based Agent	53.33	1.33	604.53	309.86	0.512
TransFuser	98.08	0.55	3661.08	658.94	0.179
LMDrive	16.59	1.32	2614.61	177.87	0.068

Table 8: Results of different models in default weather

7.1.2 Conflict Scenarios

Furthermore, we conducted experiments involving conflict scenarios (as defined in Section 6.1.2) for each model. Table 9 presents the average values of the metrics. The results indicate that although TransFuser experienced an increased number of collisions, it still maintained high-performance levels. In contrast, LMDrive maintained a consistent NOC due to its low route completion rate. The rule-based agent remained unaffected by conflicts because it relies solely on simulator back-end information, without using sensory inputs like cameras or LiDAR. These conflict scenarios demonstrate that extreme conditions significantly impact the sensor data received.

The comprehensive analysis reveals that TransFuser not only outperforms LMDrive across metrics but also shows stability in varied and challenging environments. These findings suggest that the Multi-Modal Fusion Transformer in TransFuser provides substantial advantages in efficiency and adaptability, which are critical for real-time applications in autonomous driving. Additionally, the inefficiencies in LMDrive are due to its poor architecture of Vision Encoder and integration of LLM.

Models	Conflicts	RC	NOC	ST	GT
Rule-based Agent	medium	91.58	0	895.57	623.05
	extreme	97.03	1	916.27	623.05
TransFuser	medium	100	6	2949.83	576.35
	extreme	98.84	12	3103.19	623.0
LMDrive	medium	19.04	1	3992.65	287.35
	extreme	15.31	0	1428.86	98.8

Table 9: Results of different models in conflict scenarios

7.2 Interpretability of LMDrive

In this section, we answer the research question 4, formulated in Eq.5.4. We focus on a deeper analysis of its interpretation abilities and the specific functional aspects where the model shows failure, and how these insights could inform improvements in autonomous driving technology. Although we collect data by screenshots of every change of the instructions and notifications, we choose the most representative cases to elaborate on the interpretability of LMDrive.

7.2.1 Failure of Instructions

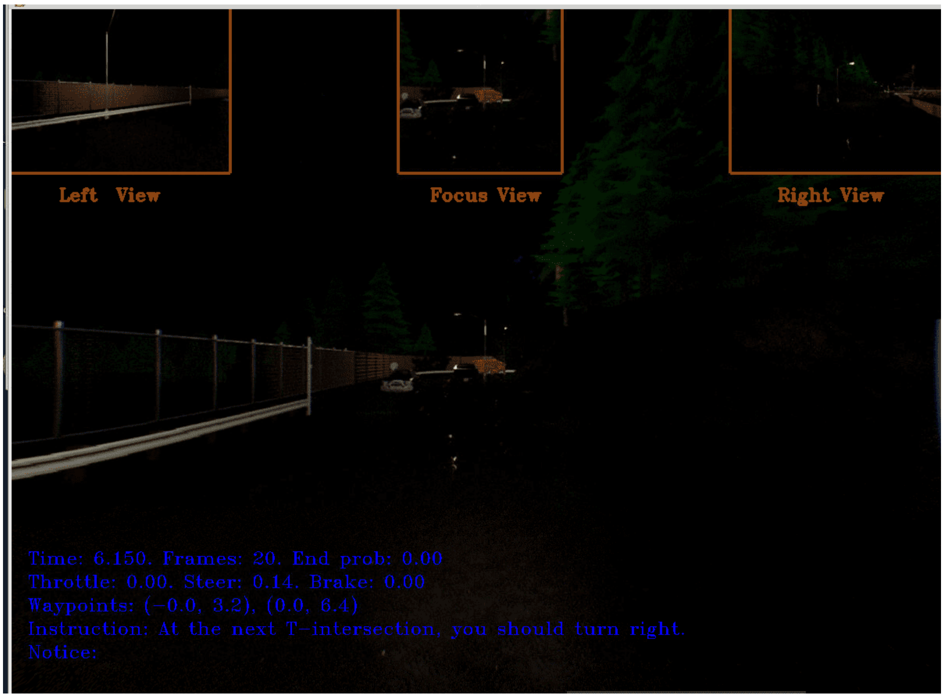
The LMDrive model has shown some shortcomings in its ability to follow instructions that are correctly aligned with the planned route, leading to critical errors such as collisions with obstacles and mis-navigation. These errors can be attributed to differences between the model's interpretation and the route planning modules(as shown in Figures 13).

We identify three types of failure: (1) The instruction is correct, but the agent fails to execute the action; (2) The instruction is incorrect, but the agent does not follow it; and (3) The instruction is incorrect, and the agent follows it. These instructions were observed in the simulation, as illustrated in Figure 10. Failures of type 1 commonly occur when the agent completes a turn. The system categorizes type 2 failures with a [Misleading] label. Type 3 represents a critical error, but it occurs infrequently Fig.13.

Failure Type	Instructions
1	Continue driving straight on this particular road Do not derive from this road Keep on rolling straight till you get to the next junction
2	[Misleading] Please adjust your course to the left-most lane In another 16 meters, you'll be turning left at the T- junction, alright? [Misleading] Just keep on rolling down the highway
3	keep going on this road, you're doing great

Table 10: Samples of failure of instructions

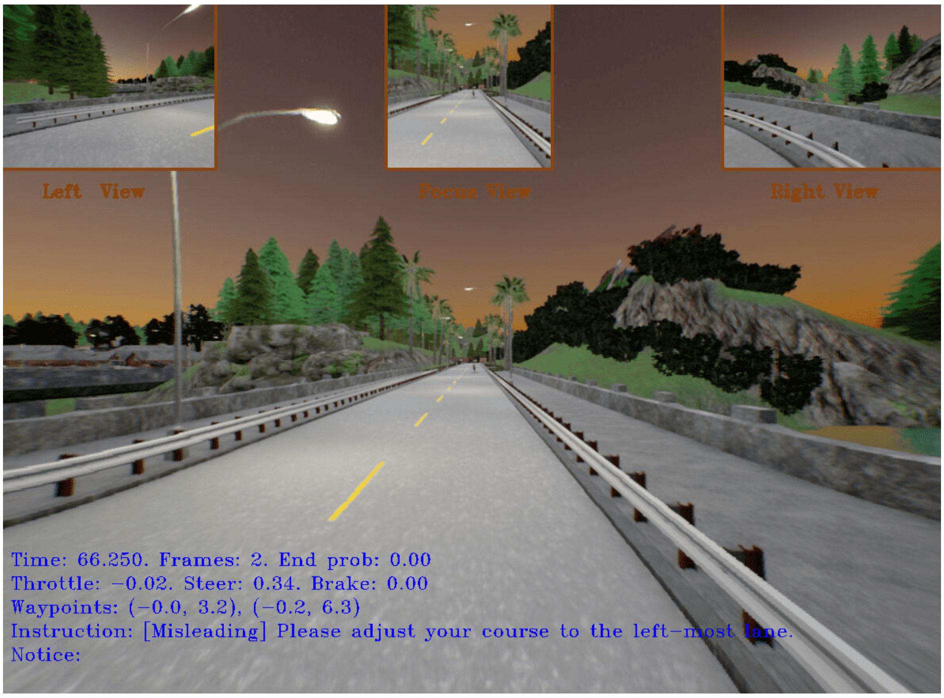
Type 1: failure occurred when the agent could not turn right at the first T-intersection, straying from the planned route despite having received accurate instructions. Figure 11 illustrates this scenario, which was tested on Route 2(see in Fig. 9b) in extreme weather conditions. However, under normal conditions, the agent successfully follows the instructions. This suggests that the model is impacted by conflicts No.2 and No.10 (see in Table 3) because road landmarks are obscured and sensors are unable to receive data due to weather interference.



```
Time: 6.150. Frames: 20. End prob: 0.00
Throttle: 0.00. Steer: 0.14. Brake: 0.00
Waypoints: (−0.0, 3.2), (0.0, 6.4)
Instruction: At the next T−intersection, you should turn right.
Notice:
```

Figure 11: The T-intersection is ahead of the agent and the system instructs to turn right

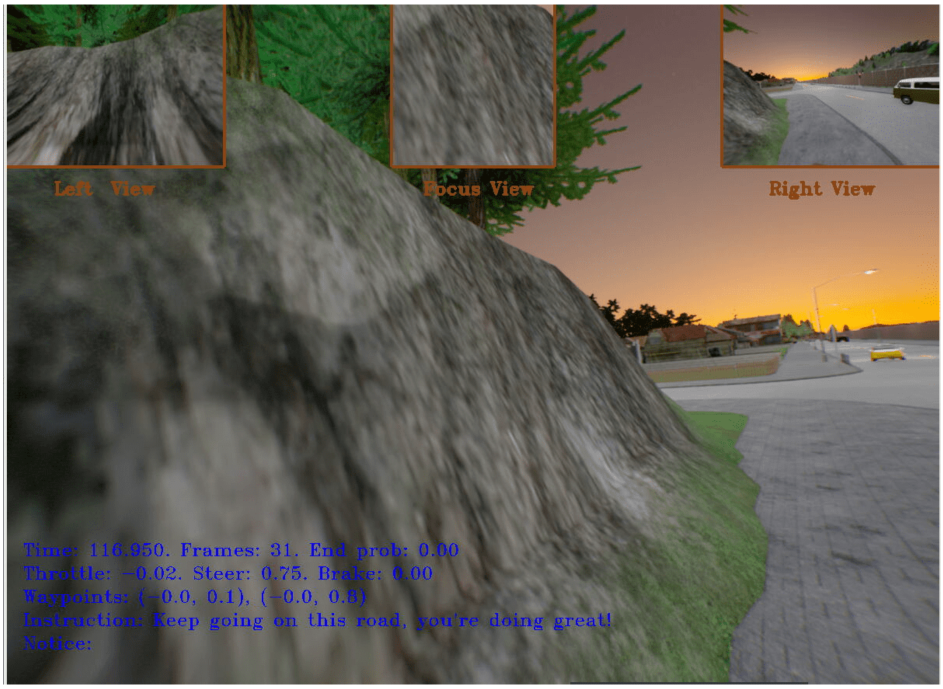
Type 2: The system provided incorrect instructions, even though the route to proceed straight was evident. This case, illustrated in Figure 12, was tested on Route 1(see in Fig.9a) under normal weather conditions. The [Misleading] label is derived from the visual feature buffer and the remaining misleading frames, suggesting that the model is capable of detecting system errors.



```
Time: 66.250. Frames: 2. End prob: 0.00
Throttle: −0.02. Steer: 0.34. Brake: 0.00
Waypoints: (−0.0, 3.2), (−0.2, 6.3)
Instruction: [Misleading] Please adjust your course to the left−most lane.
Notice:
```

Figure 12: The system instructs the agent to change course to the other lane

Type 3: The agent clearly followed incorrect instructions, resulting in a collision with a rock on the sidewalk. This case, presented in Figure 13, was tested on Route 1(see in Fig.9a) and failed at the third turn. This indicates a lack of coherence between the system's motion planning and vehicle control. The failure likely stems from the model's inability to accurately interpret spatial instructions within the context of the vehicle's current trajectory and environmental conditions.



```
Time: 116.950. Frames: 31. End prob: 0.00
Throttle: −0.02. Steer: 0.75. Brake: 0.00
Waypoints: (−0.0, 0.1), (−0.0, 0.8)
Instruction: Keep going on this road, you're doing great!
Notice:
```

Figure 13: The system instructs the agent to drive straight

7.2.2 Traffic Lights Notifications

Despite its limitations in route adherence, the LMDrive model demonstrates a notable competence in traffic light recognition and notification. The model effectively identifies and communicates the status of traffic lights (green, yellow, and red), which indicates a well-functioning component within its broader architecture. This functionality is essential for ensuring compliance with traffic regulations and enhancing road safety.

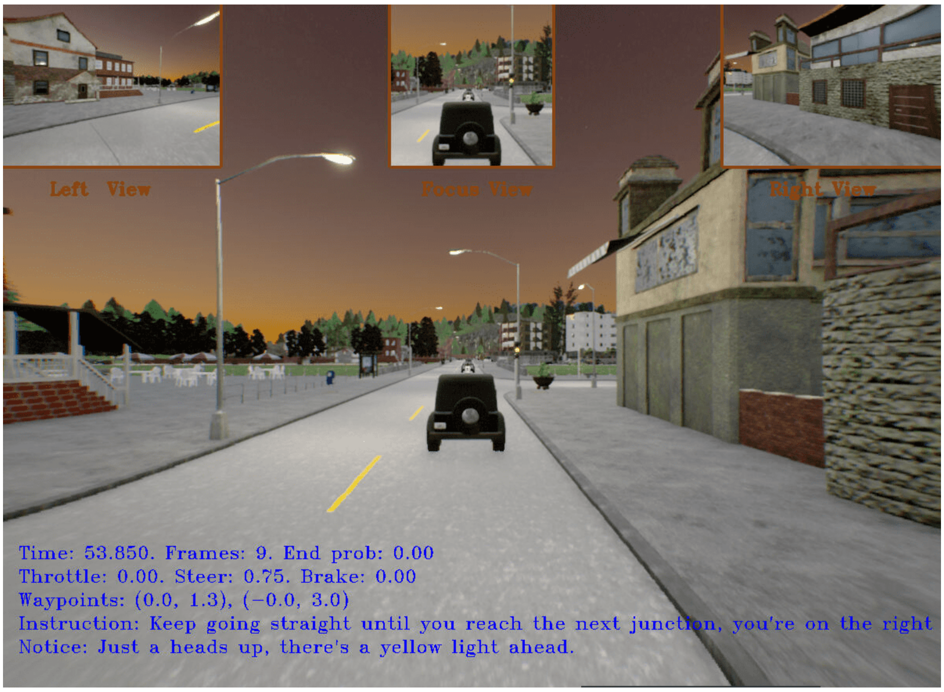
The notification samples presented in Table 11 successfully provide accurate information regarding traffic light statuses. Each route features different scenarios. For instance, Route 1 begins with a red light, transitions to a green light, and does not encounter a yellow light in a single traversal. Route 2 starts with a green light, turns right, and encounters three traffic lights. Route 3 initially faces a red light at the back of traffic, followed by another red light after leaving the lane. Similarly, it passes through a green light but only encounters a single yellow light.

We use Figure 14 as a representative example because immediately after the yellow light, the notification detects a bumpy road ahead, as shown in Figure 15. This indicates that the

Traffic Light	Route	Notification
Red	1	Just a heads up, there's a red light ahead
	2	Please be alert of the red traffic signal ahead
	3	Watch for the red light front
	3	Just a heads up, there's a red light ahead
Green	1	Just a heads up, there's a green light ahead
	2	Please be alert of the green traffic signal ahead
	3	Attention is required the green light ahead.
	3	Just a heads up, there's a green light ahead
Yellow	1	-
	2	Watch for the yellow light front
	3	-
	3	Just a heads up, there's a yellow light ahead.

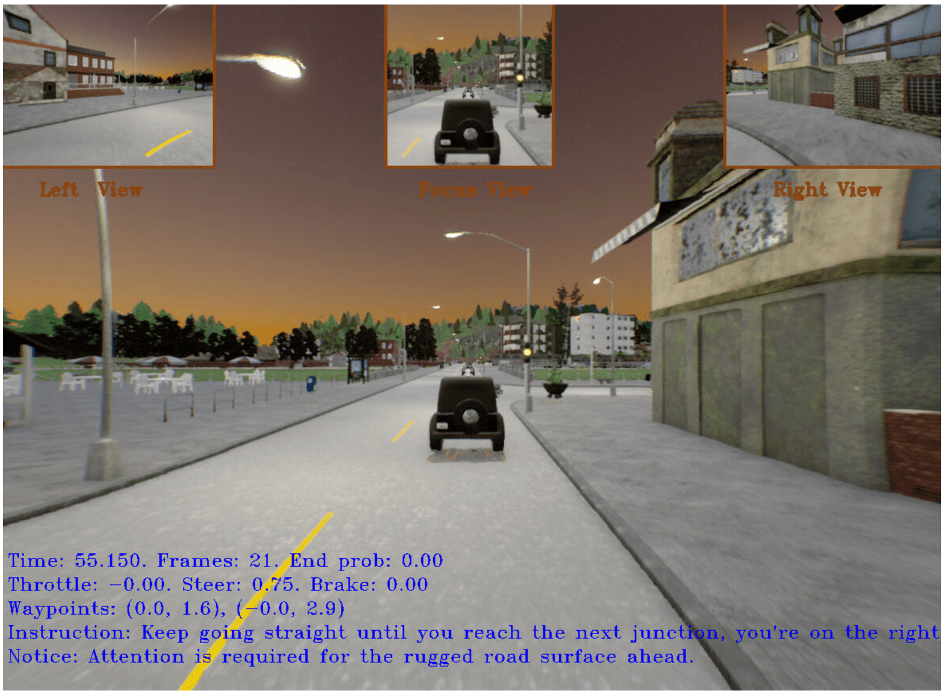
Table 11: Notifications collected from 3 routes both on default and extreme conflict.

interpretability of LMDrive integrates well with the sensor system.



```
Time: 53.850. Frames: 9. End prob: 0.00
Throttle: 0.00. Steer: 0.75. Brake: 0.00
Waypoints: (0.0, 1.3), (−0.0, 3.0)
Instruction: Keep going straight until you reach the next junction, you're on the right
Notice: Just a heads up, there's a yellow light ahead.
```

Figure 14: The system gives a notification about the yellow light ahead



```
Time: 55.150. Frames: 21. End prob: 0.00
Throttle: −0.00. Steer: 0.75. Brake: 0.00
Waypoints: (0.0, 1.6), (−0.0, 2.9)
Instruction: Keep going straight until you reach the next junction, you're on the right
Notice: Attention is required for the rugged road surface ahead.
```

Figure 15: The system asks for attention to the rugged road ahead

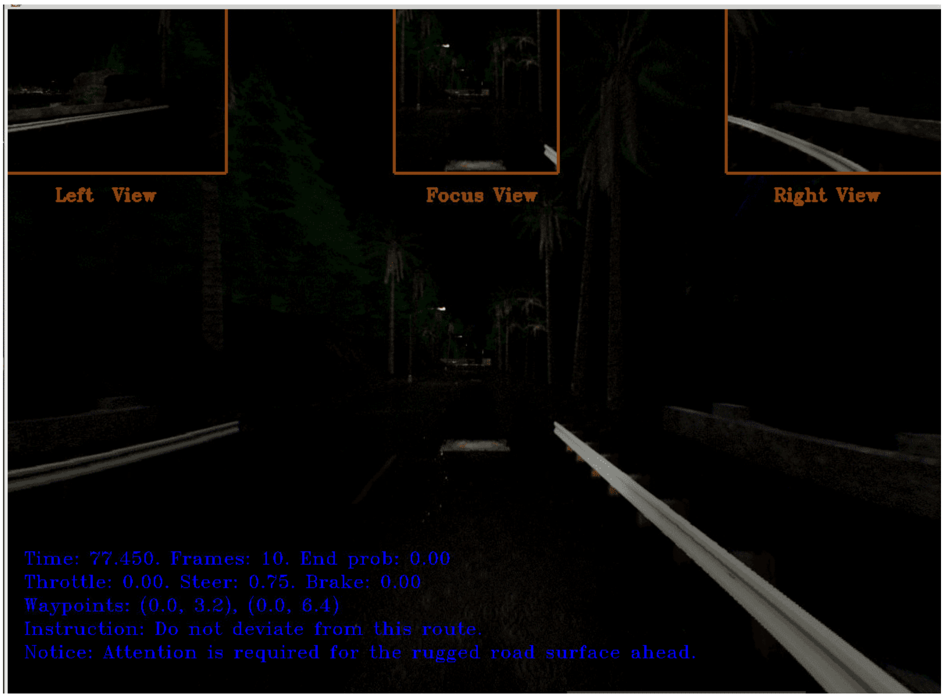
7.2.3 Bumpy Road Detection

We collected samples of the notifications triggered when the agent encountered a bumpy road, as shown in Figure 12. One example, presented in Figure 16, was selected to illustrate the No.9 testing scenarios. The results of this study indicate that the knowledge-driven model is capable of detecting bumpy roads during driving.

Notification for Bumpy Road
Attention is required for the rugged road surface head
Bumpy Road Head
Just a heads up, the road's a bit bumpy ahead
Please be alert of the uneven road surface in the vicinity ahead

Table 12: Samples of Notification for Bumpy Road

In spite of lacking lights in extreme conditions, the vehicle is able to detect a bumpy road. As Figure 16 shows the vehicle is driving in an extreme weather condition in route 1.



```
Time: 77.450. Frames: 10. End prob: 0.00
Throttle: 0.00. Steer: 0.75. Brake: 0.00
Waypoints: (0.0, 3.2), (0.0, 6.4)
Instruction: Do not deviate from this route.
Notice: Attention is required for the rugged road surface ahead.
```

Figure 16: Notice: Attention is required for the rugged road surface head

7.3 Summary

The results present the evaluation of two autonomous driving models, TransFuser and LMDrive. It compares their performance across various metrics, such as route completion (RC), number of collisions (NOC), system time (ST), game time (GT) and inference time(IT). Here are the key findings:

Performance Comparison: TransFuser significantly outperforms LMDrive, achieving an almost 100% completion rate, while LMDrive averages 16.59% across multiple routes. TransFuser takes longer system time due to its higher completion rate. LMDrive has higher inference times and also struggles with collisions and navigation errors. TransFuser shows consistent performance, with fewer collisions and greater accuracy in diverse scenarios. In conflict scenarios, TransFuser maintains high performance despite an increase in collisions, while LMDrive's completion rate drops even further. The rule-based agent remains stable due to its reliance on simulator back-end information.

Interpretability: The model sometimes fails to follow instructions correctly, leading to collisions or mis-navigation. Errors occur due to misalignment between model interpretation and the planned route. But in Traffic Lights and Road Detection, LMDrive excels at recognizing and notifying about traffic lights and road conditions, accurately detecting bumpy roads and changes in traffic signals.

8 Conclusions

We provided a comprehensive analysis of how integrating large language models (LLMs) into self-driving systems enhances interpretability and planning. We explored the implementation of both data-driven and knowledge-driven models within CARLA simulator, using frameworks such as TransFuser and LMDrive. The comparative evaluation, focusing on metrics such as route completion and collision count, highlighted the dual potential and limitations inherent in these approaches.

Through comparative analysis of different models, we discussed the strengths and weaknesses of data-driven and knowledge-driven approaches. TransFuser efficiently integrates images and LiDAR data, while LMDrive uses a blend of sensory data and language models to offer improved perception and contextual understanding.

The results revealed that while LMDrive shows limitations in motion planning, it excels in interpretability, especially in scenarios involving traffic light notifications and bumpy road detection. This suggests that LLMs can be effectively used to enhance the safety and reliability of autonomous driving systems by enhancing their ability to interpret and react to complex driving environments.

However, our study has several limitations. We focus on specific models like TransFuser and LMDrive. While these models are advanced, they may not fully represent the diversity of autonomous driving technologies available. The results might vary with different architectures or in different simulation environments. The scope of interpretability is still limited. For example, comprehensive real-world applicability requires a broader spectrum of interpretative capabilities, including more nuanced traffic interactions and rare emergency scenarios.

Our research emphasizes evaluating model interpretability and suggests that future studies should examine a wider range of LLMs and autonomous driving frameworks. This approach would aid in identifying the most efficient models for diverse driving conditions and scenarios. Finally, we will refine the framework for assessing interpretability, allowing for a more rigorous and standardized understanding of how large language models contribute to transparent and interpretable decision-making in self-driving systems.

Bibliography

Agarwal, Sandhini, Ilge Akkaya, Valerie Balcom, Mo Bavarian, Gabriel Bernadett-Shapiro, Greg Brockman, Miles Brundage, et al. March 2023. "ChatGPT plugins". *OpenAI Blog* ().

Aodha, Oisin Mac, Shihan Su, Yuxin Chen, Pietro Perona, and Yisong Yue. 2018. *Teaching Categories to Human Learners with Visual Explanations.* arXiv: 1802.06924 `[cs.CV]`.

ASAM e.V. 2023. *ASAM OpenDRIVE®.* Accessed: 2024-04-28. https://www.asam.net/standards/detail/opendrive/.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2016. *Neural Machine Translation by Jointly Learning to Align and Translate.* arXiv: 1409.0473 `[cs.CL]`.

Bai, Jinze, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. "Qwen-VL: A Versatile Vision-Language Model for Understanding, Localization, Text Reading, and Beyond". *arXiv preprint arXiv:2308.12966.*

Basha, SH Shabbeer, Shiv Ram Dubey, Viswanath Pulabaigari, and Snehasis Mukherjee. 2020. "Impact of fully connected layers on performance of convolutional neural networks for image classification". *Neurocomputing* 378:112–119.

Blanco, Myra, Jon Atwood, Holland Vasquez, Tammy Trimble, Vikki Fitchett, Joshua Radlbeck, Gregory Fitch, et al. August 2015. *Human Factors Evaluation of Level 2 and Level 3 Automated Driving Concepts.* https://doi.org/10.13140/RG.2.1.1874.7361.

Bose, Bimal K. 2007. "Neural network applications in power electronics and motor drives—An introduction and perspective". *IEEE Transactions on Industrial Electronics* 54 (1): 14–33.

Brohan, Anthony, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, et al. 2023. *RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control.* arXiv: 2307.15818 `[cs.RO]`.

Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, et al. 2020. *Language Models are Few-Shot Learners.* arXiv: 2005.14165 `[cs.CL]`.

Caesar, Holger, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. 2020. *nuScenes: A multimodal dataset for autonomous driving.* arXiv: 1903.11027 `[cs.LG]`.

Cao, Zhong, Xiang Li, Kun Jiang, Weitao Zhou, Xiaoyu Liu, Nanshan Deng, and Diange Yang. 2023. "Autonomous Driving Policy Continual Learning With One-Shot Disengagement Case". *IEEE Transactions on Intelligent Vehicles* 8 (2): 1380–1391. https://doi.org/10.1109/TIV.2022.3184729.

Carion, Nicolas, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. *End-to-End Object Detection with Transformers.* arXiv: 2005.12872 `[cs.CV]`.

Chen, Li, Penghao Wu, Kashyap Chitta, Bernhard Jaeger, Andreas Geiger, and Hongyang Li. 2023. *End-to-end Autonomous Driving: Challenges and Frontiers.* arXiv: 2306.16927 `[cs.RO]`.

Chen, Long, Yuchen Li, Chao Huang, Yang Xing, Daxin Tian, Li Li, Zhongxu Hu, et al. 2023. "Milestones in Autonomous Driving and Intelligent Vehicles—Part I: Control, Computing System Design, Communication, HD Map, Testing, and Human Behaviors". *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 53 (9): 5831–5847. https://doi.org/10.1109/TSMC.2023.3276218.

Chen, Long, Oleg Sinavski, Jan Hünermann, Alice Karnsund, Andrew James Willmott, Danny Birch, Daniel Maund, and Jamie Shotton. 2023. *Driving with LLMs: Fusing Object-Level Vector Modality for Explainable Autonomous Driving.* arXiv: 2310.01957 `[cs.RO]`.

Chitta, Kashyap, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. 2022. *TransFuser: Imitation with Transformer-Based Sensor Fusion for Autonomous Driving.* arXiv: 2205.15997 `[cs.CV]`.

Cho, Kyunghyun, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.* arXiv: 1406.1078 `[cs.CL]`.

Choi, Jong, and Yong Gu Ji. July 2015. “Investigating the Importance of Trust on Adopting an Autonomous Vehicle”. *International Journal of Human-Computer Interaction* 31 (): 150709133142005. https://doi.org/10.1080/10447318.2015.1070549.

Choudhary, Tushar, Vikrant Dewangan, Shivam Chandhok, Shubham Priyadarshan, Anushka Jain, Arun K. Singh, Siddharth Srivastava, Krishna Murthy Jatavallabhula, and K. Madhava Krishna. 2023. *Talk2BEV: Language-enhanced Bird’s-eye View Maps for Autonomous Driving.* arXiv: 2310.02251 `[cs.CV]`.

Chung, Junyoung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling.* arXiv: 1412.3555 `[cs.NE]`.

Commons, Creative. 2022. *Creative Commons.* https://creativecommons.org. Accessed: 2024-05-01.

Crayton, Travis J, and Benjamin Mason Meier. 2017. “Autonomous vehicles: Developing a public health research agenda to frame the future of transportation policy”. *Journal of Transport & Health* 6:245–252.

Cui, Can, Yunsheng Ma, Xu Cao, Wenqian Ye, and Ziran Wang. 2023. *Receive, Reason, and React: Drive as You Say with Large Language Models in Autonomous Vehicles.* arXiv: 2310.08034 `[cs.HC]`.

Cui, Can, Zichong Yang, Yupeng Zhou, Yunsheng Ma, Juanwu Lu, Lingxi Li, Yaobin Chen, Jitesh Panchal, and Ziran Wang. 2024. *Personalized Autonomous Driving with Large Language Models: Field Experiments.* arXiv: 2312.09397 `[cs.AI]`.

Dai, Wenliang, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. 2023. *InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning.* arXiv: 2305.06500 `[cs.CV]`.

Developer, E., and F. Researcher. 2023. “Generative Pre-trained Transformer Models for Multi-Modal and Multi-Task Learning”. *Journal of AI Research* 55 (1): 99–110.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* arXiv: 1810.04805 `[cs.CL]`.

———. 2019b. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In *Proceedings of the 2019 Conference of the Association for Computational Linguistics: Human Language Technologies, ACL 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1: Long Papers,* 4171–4186. Association for Computational Linguistics. https://www.aclweb.org/anthology/P19-1423/.

Documentation, Simulink. 2020. *Simulation and Model-Based Design.* https://www.mathworks.com/products/simulink.html.

Dolgov, Dmitri, Sebastian Thrun, Michael Montemerlo, and James Diebel. 2008. “Practical search techniques in path planning for autonomous driving”. *Ann Arbor* 1001 (48105): 18–80.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, et al. 2021. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.* arXiv: 2010.11929 `[cs.CV]`.

Dosovitskiy, Alexey, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. *CARLA: An Open Urban Driving Simulator.* arXiv: 1711.03938 `[cs.LG]`.

Dreissig, Mariella, Dominik Scheuble, Florian Piewak, and Joschka Boedecker. 2023. *Survey on LiDAR Perception in Adverse Weather Conditions.* arXiv: 2304.06312 `[cs.RO]`.

Eldabi, Tillal, Zahir Irani, and Ray Paul. February 2002. “Quantitative and qualitative decision-making methods in simulation modeling”. *Management Decision - MANAGE DECISION* 40 (): 64–73. https://doi.org/10.1108/00251740210413370.

Fang, Yuxin, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. 2022. *EVA: Exploring the Limits of Masked Visual Representation Learning at Scale.* arXiv: 2211.07636 `[cs.CV]`.

Fitzgibbons, J.B., R.M. Fujimoto, D. Fellig, S.D. Kleban, and A.J. Scholand. 2004. “IDSim: an extensible framework for Interoperable Distributed Simulation”. In *Proceedings. IEEE International Conference on Web Services, 2004.* 532–539. https://doi.org/10.1109/ICWS.2004.1314779.

Fu, Daocheng, Xin Li, Licheng Wen, Min Dou, Pinlong Cai, Botian Shi, and Yu Qiao. 2023. *Drive Like a Human: Rethinking Autonomous Driving with Large Language Models.* arXiv: 2307.07162 `[cs.RO]`.

Gold, Christian, Frederik Naujoks, Jonas Radlmayr, Hanna Otto, and Oliver Jarosch. June 2017. “Testing Scenarios for Human Factors Research in Level 3 Automated Vehicles”, 551–559. ISBN: 978-3-319-60440-4. https://doi.org/10.1007/978-3-319-60441-1_54.

Grigorescu, Sorin, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. 2020. “A survey of deep learning techniques for autonomous driving”. *Journal of field robotics* 37 (3): 362–386.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. *Deep Residual Learning for Image Recognition.* arXiv: 1512.03385 `[cs.CV]`.

———. 2016. “Deep residual learning for image recognition”. In *Proceedings of the IEEE conference on computer vision and pattern recognition,* 770–778.

Hecker, Simon, Dengxin Dai, Alexander Liniger, and Luc Van Gool. 2020. *Learning Accurate and Human-Like Driving using Semantic Maps and Attention.* arXiv: 2007.07218 `[cs.CV]`.

Hochreiter, Sepp, and Jürgen Schmidhuber. December 1997a. “Long Short-term Memory”. *Neural computation* 9 (): 1735–80. https://doi.org/10.1162/neco.1997.9.8.1735.

———. November 1997b. “Long Short-Term Memory”. *Neural Computation* 9, number 8 (): 1735–1780. ISSN: 0899-7667. https://doi.org/10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. https://doi.org/10.1162/neco.1997.9.8.1735.

Hu, Anthony, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. 2023. *GAIA-1: A Generative World Model for Autonomous Driving.* arXiv: 2309.17080 `[cs.CV]`.

Huang, Siyuan, Bo Zhang, Botian Shi, Peng Gao, Yikang Li, and Hongsheng Li. 2023. *SUG: Single-dataset Unified Generalization for 3D Point Cloud Classification.* arXiv: 2305.09160 `[cs.CV]`.

Innovator, I., and J. Thinker. 2024. *ShareGPT: Sharing Large-Scale Generative Models.* https://sharegpt.example.com.

Jain, Kanishk, Varun Chhangani, Amogh Tiwari, K. Madhava Krishna, and Vineet Gandhi. 2022. *Ground then Navigate: Language-guided Navigation in Dynamic Scenes.* arXiv: 2209.11972 `[cs.CV]`.

Jin, Ye, Xiaoxi Shen, Huiling Peng, Xiaoan Liu, Jingli Qin, Jiayang Li, Jintao Xie, Peizhong Gao, Guyue Zhou, and Jiangtao Gong. 2023. *SurrealDriver: Designing Generative Driver Agent Simulation Framework in Urban Contexts based on Large Language Model.* arXiv: 2309.13193 `[cs.HC]`.

Johansson, Richard, David Williams, Anders Berglund, and Pierre Nugues. 2004. “Carsim: a system to visualize written road accident reports as animated 3D scenes”. In *Proceedings of the 2nd Workshop on Text Meaning and Interpretation,* 57–64. TextMean ’04. Barcelona, Spain: Association for Computational Linguistics.

Kim, Jinkyu, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. 2018. *Textual Explanations for Self-Driving Vehicles.* arXiv: 1807.11546 `[cs.CV]`.

Kolesnikov, Alexander, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. 2020. *Big Transfer (BiT): General Visual Representation Learning.* arXiv: 1912.11370 `[cs.CV]`.

Lang, Alex H, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. 2019. “PointPillars: Fast encoders for object detection from point clouds”. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition,* 12697–12705.

Law, Averill M., and W. David Kelton. 1991. *Simulation Modeling and Analysis.* 2nd edition. McGraw-Hill.

Leaderboard, CARLA. 2024. *CARLA Leaderboard.* https://github.com/carla-simulator/leaderboard. Accessed: 2024-05-06.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep learning". *nature* 521 (7553): 436–444.

Lee, John, and Neville Moray. 1992. "Trust, control strategies and allocation of function in human-machine systems". *Ergonomics* 35 (10): 1243–1270. https://doi.org/10.1080/00140139208967392.

Leurent, Edouard. 2018. *An Environment for Autonomous Driving Decision-Making.* https://github.com/eleurent/highway-env.

Li, Junnan, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. *BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models.* arXiv: 2301.12597 `[cs.CV]`.

Li, Ning, Lang Hu, Zhong-Liang Deng, Tong Su, and Jiang-Wang Liu. May 2021. "Research on GRU Neural Network Satellite Traffic Prediction Based on Transfer Learning". *Wireless Personal Communications* 118, number 1 (): 815–827. https://doi.org/10.1007/s11277-020-08045-z. https://doi.org/10.1007/s11277-020-08045-z.

Li, You, and Javier Ibanez-Guzman. 2020. "Lidar for Autonomous Driving: The Principles, Challenges, and Trends for Automotive Lidar and Perception Systems". *IEEE Signal Processing Magazine* 37 (4): 50–61. https://doi.org/10.1109/MSP.2020.2973615.

Lin, Chin-Yew. July 2004. "ROUGE: A Package for Automatic Evaluation of Summaries". In *Text Summarization Branches Out,* 74–81. Barcelona, Spain: Association for Computational Linguistics. https://aclanthology.org/W04-1013.

Liu, Hao Tian. 2023. *llava-v1.5-7b.* https://huggingface.co/liuhaotian/llava-v1.5-7b.

Liu, Haotian, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2023. *Improved Baselines with Visual Instruction Tuning.* arXiv: 2310.03744 `[cs.CV]`.

Liu, Haotian, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. *Visual Instruction Tuning.* arXiv: 2304.08485 `[cs.CV]`.

Liu, Jiaqi, Peng Hang, Xiao qi, Jianqiang Wang, and Jian Sun. 2023. *MTD-GPT: A Multi-Task Decision-Making GPT Model for Autonomous Driving at Unsignalized Intersections.* arXiv: 2307.16118 `[cs.RO]`.

Mao, Jiageng, Yuxi Qian, Junjie Ye, Hang Zhao, and Yue Wang. 2023. *GPT-Driver: Learning to Drive with GPT.* arXiv: 2310.01415 `[cs.CV]`.

Mao, Jiageng, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. 2023. *A Language Agent for Autonomous Driving.* arXiv: 2311.10813 `[cs.CV]`.

Miller, Tim. 2018. *Explanation in Artificial Intelligence: Insights from the Social Sciences.* arXiv: 1706.07269 `[cs.AI]`.

OpenAI. 2023. *Introducing ChatGPT.* https://openai.com/blog/chatgpt/.

OpenDILab Community. 2023. *LMDrive LLaMA-v1.5-7b v1.0.* https://huggingface.co/OpenDILabCommunity/LMDrive-llava-v1.5-7b-v1.0.

Ordonez, Vicente, Girish Kulkarni, and Tamara L. Berg. 2011. “The SBU Captioned Photo Dataset”. In *Proceedings of the 2011 IEEE International Conference on Computer Vision,* 2537–2544. IEEE.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. “Bleu: a method for automatic evaluation of machine translation”. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics,* 311–318.

Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio. 2013. *On the difficulty of training Recurrent Neural Networks.* arXiv: 1211.5063 `[cs.LG]`.

Patil, Priyadarshan. 2020. “The Future of Electric Vehicles: A Comprehensive Review of Technological Advancements, Market Trends, and Environmental Impacts”. *Journal of Artificial Intelligence and Machine Learning in Management* 4 (1): 56–68.

Paul, Ray J, and David W Balmer. 1993. *Simulation modelling.* Chartwell-Bratt.

Prakash, Aditya, Kashyap Chitta, and Andreas Geiger. 2021. *Multi-Modal Fusion Transformer for End-to-End Autonomous Driving.* arXiv: 2104.09224 `[cs.CV]`.

Qi, Charles R, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. 2018. “Frustum Pointnets for 3d object detection from RGB-D data”. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 918–927.

Quinn, J.B., I.Y. Mintzberg, and R.M. James. 1988. “The Strategic Process, Concepts, Context and Cases”. In *The Strategic Process, Concepts, Context and Cases,* edited by J.B. Quinn, I.Y. Mintzberg, and R.M. James. Englewood Cliffs, NJ: Prentice-Hall.

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, et al. 2021. *Learning Transferable Visual Models From Natural Language Supervision.* arXiv: 2103.00020 `[cs.CV]`.

Raitoharju, Jenni. 2022. “Chapter 3 - Convolutional neural networks”. In *Deep Learning for Robot Perception and Cognition,* edited by Alexandros Iosifidis and Anastasios Tefas, 35–69. Academic Press. ISBN: 978-0-323-85787-1. https://doi.org/https://doi.org/10.1016/B978-0-32-385787-1.00008-7. https://www.sciencedirect.com/science/article/pii/B9780323857871000087.

Ravanelli, Mirco, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. April 2018. “Light Gated Recurrent Units for Speech Recognition”. *IEEE Transactions on Emerging Topics in Computational Intelligence* 2, number 2 (): 92–102. ISSN: 2471-285X. https://doi.org/10.1109/tetci.2017.2762739. http://dx.doi.org/10.1109/TETCI.2017.2762739.

S.O.-R.A.V.S. Committee. 2021. “Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems”, https://www.sae.org/standards/content/j3016_202104/.

ScenarioRunner, CARLA. 2024. *CARLA ScenarioRunner.* https://github.com/carla-simulator/scenario_runner. Accessed: 2024-05-06.

Schuhmann, Christoph, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, et al. 2022. *LAION-5B: An open large-scale dataset for training next generation image-text models.* arXiv: 2210.08402 `[cs.CV]`.

Scientist, G., and H. Analyst. 2017. “Visual Question Answering: A Survey of Methods and Datasets”. *Computer Vision and Image Understanding* 163:21–40.

Sha, Hao, Yao Mu, Yuxuan Jiang, Li Chen, Chenfeng Xu, Ping Luo, Shengbo Eben Li, Masayoshi Tomizuka, Wei Zhan, and Mingyu Ding. 2023. *LanguageMPC: Large Language Models as Decision Makers for Autonomous Driving.* arXiv: 2310.03026 `[cs.RO]`.

Shah, Dhruv, Blazej Osinski, Brian Ichter, and Sergey Levine. 2022. *LM-Nav: Robotic Navigation with Large Pre-Trained Models of Language, Vision, and Action.* arXiv: 2207.04429 `[cs.RO]`.

Shannon, R.E. 1998. “Introduction to the art and science of simulation”. In *1998 Winter Simulation Conference. Proceedings (Cat. No.98CH36274),* volume 1, 7–14 vol.1. https://doi.org/10.1109/WSC.1998.744892.

Shao, Hao, Yuxuan Hu, Letian Wang, Steven L. Waslander, Yu Liu, and Hongsheng Li. 2023. *LMDrive: Closed-Loop End-to-End Driving with Large Language Models.* arXiv: 2312.07488 `[cs.CV]`.

Sharma, Sagar, Simone Sharma, and Anidhya Athaiya. 2017. “Activation functions in neural networks”. *Towards Data Sci* 6 (12): 310–316.

Shen, Yuan, Shanduojiao Jiang, Yanlin Chen, and Katie Driggs Campbell. 2022. *To Explain or Not to Explain: A Study on the Necessity of Explanations for Autonomous Vehicles.* arXiv: 2006.11684 `[cs.AI]`.

Sima, Chonghao, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Ping Luo, Andreas Geiger, and Hongyang Li. 2023. *DriveLM: Driving with Graph Visual Question Answering.* arXiv: 2312.14150 `[cs.CV]`.

Singh, Santokh. 2015. *Critical reasons for crashes investigated in the national motor vehicle crash causation survey.* Technical report.

Sun, Pei, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, et al. 2020. *Scalability in Perception for Autonomous Driving: Waymo Open Dataset.* arXiv: 1912.04838 `[cs.CV]`.

Taeihagh, Araz, and Hazel Si Min Lim. July 2018. “Governing autonomous vehicles: emerging responses for safety, liability, privacy, cybersecurity, and industry risks”. *Transport Reviews* 39, number 1 (): 103–128. ISSN: 1464-5327. https://doi.org/10.1080/01441647.2018.1494640. http://dx.doi.org/10.1080/01441647.2018.1494640.

Tammewar, Akshaj, Nikita Chaudhari, Bunny Saini, Divya Venkatesh, Ganpathiraju Dharahas, Deepali Vora, Shruti Patil, Ketan Kotecha, and Sultan Alfarhood. 2023. “Improving the Performance of Autonomous Driving through Deep Reinforcement Learning”. *Sustainability* 15 (18). ISSN: 2071-1050. https://doi.org/10.3390/su151813799. https://www.mdpi.com/2071-1050/15/18/13799.

Techie, C., and D. Designer. 2022. “BLIP: Blogging Language Image Processor”. In *Proceedings of the International Conference on Multimedia Retrieval.* New York, NY, USA: ACM.

Thrun, Sebastian. April 2010. “Toward robotic cars”. *Commun. ACM* (New York, NY, USA) 53, number 4 (): 99–106. ISSN: 0001-0782. https://doi.org/10.1145/1721654.1721679. https://doi.org/10.1145/1721654.1721679.

Tian, Xiaoyu, Junru Gu, Bailin Li, Yicheng Liu, Chenxu Hu, Yang Wang, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. 2024. *DriveVLM: The Convergence of Autonomous Driving and Large Vision-Language Models.* arXiv: 2402.12289 `[cs.CV]`.

Tian, Yuchi, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. *DeepTest: Automated Testing of Deep-Neural-Network-driven Autonomous Cars.* arXiv: 1708.08559 `[cs.SE]`.

Tom M. Gasser, Daniel Westhoff. August 2012. *BASt-study: Definitions of Automation and Legal Issues in Germany.* https://onlinepubs.trb.org/onlinepubs/conferences/2012/Automation/presentations/Gasser.pdf.

Touvron, Hugo, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, et al. 2023. *LLaMA: Open and Efficient Foundation Language Models.* arXiv: 2302.13971 `[cs.CL]`.

Touvron, Hugo, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. “Llama 2: Open foundation and fine-tuned chat models”. *arXiv preprint arXiv:2307.09288.*

Udugama, B. 2023. *Review of Deep Reinforcement Learning for Autonomous Driving.* arXiv: 2302.06370 `[cs.RO]`.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. *Attention Is All You Need.* arXiv: 1706.03762 `[cs.CL]`.

Vedantam, Ramakrishna, C. Lawrence Zitnick, and Devi Parikh. 2015. *CIDEr: Consensus-based Image Description Evaluation.* arXiv: 1411.5726 `[cs.CV]`.

Wang, Huiyu, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. 2020. *Axial-DeepLab: Stand-Alone Axial-Attention for Panoptic Segmentation.* arXiv: 2003.07853 `[cs.CV]`.

Wang, Jiangong, Xiao Wang, Tianyu Shen, Yutong Wang, Li Li, Yonglin Tian, Hui Yu, et al. 2022. “Parallel Vision for Long-Tail Regularization: Initial Results From IVFC Autonomous Driving Testing”. *IEEE Transactions on Intelligent Vehicles* 7 (2): 286–299. https://doi.org/10.1109/TIV.2022.3145035.

Wang, Shiyi, Yuxuan Zhu, Zhiheng Li, Yutong Wang, Li Li, and Zhengbing He. 2023. “ChatGPT as Your Vehicle Co-Pilot: An Initial Attempt”. *IEEE Transactions on Intelligent Vehicles* 8 (12): 4706–4721. https://doi.org/10.1109/TIV.2023.3325300.

Wang, Wenguan, Yi Yang, and Fei Wu. 2023. *Towards Data-and Knowledge-Driven Artificial Intelligence: A Survey on Neuro-Symbolic Computing.* arXiv: 2210.15889 `[cs.AI]`.

Wang, Wenhai, Jiangwei Xie, ChuanYang Hu, Haoming Zou, Jianan Fan, Wenwen Tong, Yang Wen, et al. 2023. *DriveMLM: Aligning Multi-Modal Large Language Models with Behavioral Planning States for Autonomous Driving.* arXiv: 2312.09245 `[cs.CV]`.

Wang, Xiaolong, Ross Girshick, Abhinav Gupta, and Kaiming He. 2018. *Non-local Neural Networks.* arXiv: 1711.07971 `[cs.CV]`.

Wang, Yixuan, Ruochen Jiao, Sinong Simon Zhan, Chengtian Lang, Chao Huang, Zhaoran Wang, Zhuoran Yang, and Qi Zhu. 2024. *Empowering Autonomous Driving with Large Language Models: A Safety Perspective.* arXiv: 2312.00812 `[cs.AI]`.

Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter brian, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models". In *Advances in Neural Information Processing Systems,* edited by S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, 35:24824–24837. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

Wen, Licheng, Daocheng Fu, Xin Li, Xinyu Cai, Tao Ma, Pinlong Cai, Min Dou, Botian Shi, Liang He, and Yu Qiao. 2024. *DiLu: A Knowledge-Driven Approach to Autonomous Driving with Large Language Models.* arXiv: 2309.16292 `[cs.RO]`.

Xiang, Chao, Chen Feng, Xiaopo Xie, Botian Shi, Hao Lu, Yisheng Lv, Mingchuan Yang, and Zhendong Niu. 2023. "Multi-Sensor Fusion and Cooperative Perception for Autonomous Driving: A Review". *IEEE Intelligent Transportation Systems Magazine* 15 (5): 36–58. https://doi.org/10.1109/MITS.2023.3283864.

Xie, Qizhe, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. 2020. *Self-training with Noisy Student improves ImageNet classification.* arXiv: 1911.04252 `[cs.LG]`.

Xie, Songtao, Junyan Hu, Parijat Bhowmick, Zhengtao Ding, and Farshad Arvin. 2022. "Distributed Motion Planning for Safe Autonomous Vehicle Overtaking via Artificial Potential Field". *IEEE Transactions on Intelligent Transportation Systems* 23 (11): 21531–21547. https://doi.org/10.1109/TITS.2022.3189741.

Xie, Songtao, Junyan Hu, Zhengtao Ding, and Farshad Arvin. 2023. "Cooperative Adaptive Cruise Control for Connected Autonomous Vehicles Using Spring Damping Energy Model". *IEEE Transactions on Vehicular Technology* 72 (3): 2974–2987. https://doi.org/10.1109/TVT.2022.3218575.

Xu, Zhenhua, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kwan-Yee. K. Wong, Zhenguo Li, and Hengshuang Zhao. 2024. *DriveGPT4: Interpretable End-to-end Autonomous Driving via Large Language Model.* arXiv: 2310.01412 `[cs.CV]`.

Yang, Honghui, Tong He, Jiaheng Liu, Hua Chen, Boxi Wu, Binbin Lin, Xiaofei He, and Wanli Ouyang. 2023. *GD-MAE: Generative Decoder for MAE Pre-training on LiDAR Point Clouds.* arXiv: 2212.03010 `[cs.CV]`.

Yang, Zhenjie, Xiaosong Jia, Hongyang Li, and Junchi Yan. 2023. *LLM4Drive: A Survey of Large Language Models for Autonomous Driving.* arXiv: 2311.01043 `[cs.AI]`.

Zablocki, Éloi, Hédi Ben-Younes, Patrick Pérez, and Matthieu Cord. 2022. *Explainability of deep vision-based autonomous driving systems: Review and challenges.* arXiv: 2101.05307 `[cs.CV]`.

Zhang, Junping, Jian Pu, Jie Chen, Haolin Fu, Yongzhi Tao, Shihmin Wang, Qi Chen, et al. 2023. "DSiV: Data Science for Intelligent Vehicles". *IEEE Transactions on Intelligent Vehicles* 8 (4): 2628–2634. https://doi.org/10.1109/TIV.2023.3264601.

Zhang, Qiaoning, X. Jessie Yang, and Lionel Peter Robert. 2020. "Expectations and Trust in Automated Vehicles". In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems,* 1–9. CHI EA '20. , Honolulu, HI, USA, Association for Computing Machinery. ISBN: 9781450368193. https://doi.org/10.1145/3334480.3382986. https://doi.org/10.1145/3334480.3382986.

Zhu, Deyao, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2023. *MiniGPT-4: Enhancing Vision-Language Understanding with Advanced Large Language Models.* arXiv: 2304.10592 `[cs.CV]`.

Appendices

Following the guidelines provided by GPTDriver(Mao, Qian, et al. 2023), an experiment was conducted using the CARLA Simulator(Dosovitskiy et al. 2017). The results reveal that GPT-3.5 was unable to generate appropriate waypoints, whereas GPT-4 demonstrated improved performance. However, due to the lack of detailed map data, GPT-4 was unable to effectively control the agent within the simulation environment.

System Message:

Autonomous Driving Planner
Role: You are the brain of an autonomous vehicle. Plan a safe 3-second driving trajectory. Avoid collisions with other objects.
Context
- Map:
- Coordinates: X-axis is alone with west to east, and Y-axis is alone with north to south. unit in meters.
- Objective: Create a 3-second route using 6 waypoints, one every 0.5 seconds.

Inputs
1. Perception & Prediction: Info about surrounding objects
2. Historical Trajectory: Your past 2-second route, given by 4 waypoints.
3. Ego-States: Your current state including velocity, heading angular velocity(yaw),acceleration and steering.
4. Destination Location.

Task
- Thought Process: Note down critical objects and potential effects from your perceptions and predictions.
- Action Plan: Detail your meta-actions based on your analysis.
- Trajectory Planning: Develop a safe and feasible 3-second route using 6 new waypoints.

Output
- Thoughts:
- Notable Objects

Potential Effects
- Meta Action
Please output the trajectory in the following format (MOST IMPORTANT):
- [(x1,y1), (x2,y2), (x3,y3),(x4,y4),(x5,y5),(x6,y6)]

User Message:

Perception and Prediction: - obstacle at () Historical Trajectory (last 2 seconds): - [['(-16.00,140.00)', '(-16.00,140.00)']] Ego-States: - Velocity (vx,vy): ((0.00,0.00)) - Heading Angular Velocity (yaw): ((0.00)) - Acceleration (ax,ay): ((0.00,0.00)) - Steering:(0.00) Destination:Location(x=-41.496914, y=110.004211, z=0.000000)

GPT3.5 Response:

Thoughts:
- Notable Objects:
- Obstacle at (-16.00,140.00)
- Potential Effects: Possible collision if trajectory not adjusted
Meta Action:
- Slow down or change trajectory to avoid collision with obstacle
Trajectory:- [(-16.00,140.00), (-16.00,140.00), (-16.00,140.00), (-16.00,140.00), (-16.00,140.00), (-16.00,140.00)]

GPT4 Response:

Thoughts
- Notable Objects: There's an obstacle at an unnamed location.
- Potential Effects: The obstacle could interfere with the path to the destination, potentially causing a collision.
Meta Action:
1. We must avoid the obstacle by altering the trajectory course while maintaining a high level of comfort for the occupants of the vehicle. That might mean decreasing the speed and slightly adjusting the steering angle to safely alter the trajectory around the obstruction and

then gradually align the ego vehicle back to the destination's direction.

Trajectory Planning:

The current position is (-16.00, 140.00) and the destination is (-41.496914, 110.004211). Given our current speed, trajectory and the available information, we might plan the following waypoints:- (-17.00, 135.00)- (-18.50, 130.00)- (-22.00, 125.00)- (-28.00, 120.00)- (-34.00, 115.00)- (-40.00, 110.00), Each point represents the vehicle's position every 0.5 seconds for the next 3 seconds. These values might vary depending on the size and position of the obstacle detected, the vehicle's speed and acceleration, and the overall road conditions.