CHALLENGES AND SOLUTIONS OF AUTONOMOUS DRIVING APPROACHES: A REVIEW

Samer Attrah *
Automotive and Engineering Academy
Hogeschool van Arnhem en Nijmegen
Arnhem, Gelderland, The Netherlands
samiratra95@gmail.com

April 22, 2025

ABSTRACT

In this research, we will discuss the three basic approaches to building autonomous driving systems, namely modular pipeline, end-to-end, and large models for language, vision and multi-modal models. focusing on the challenges and shortcomings of each approach and how they are solved by another, then presenting several in-depth reviews and summaries focused on the system architecture of example systems built using large models, delivering superior performance, and solving the problems of the previous two approaches. In addition to a short analysis of the most used models and datasets in developing autonomous driving systems, besides other aspects of the reviewed systems.

***K*eywords** Robotics · Automation · Artificial intelligence · Computer vision · Natural language

1 Introduction

Recently, autonomous driving has become one of the most complex studies in artificial intelligence (AI) and robotics, where an autonomous vehicle can be interpreted as a robot that needs to be in motion with precise positioning with respect to the road lanes, and other objects, besides the navigation and autonomous driving system readings. From that perspective, an autonomous driving application can take one of three forms as found in the literature: simulation, a Proof of Concept, or a real-life vehicle.

The concept of automating vehicles has been developed for a long time now, and when it started, it had nothing to do with AI, but starting from the driver without assistance, it has improved to having an optional acceleration and braking control, which was referred to as Adaptive cruise control. and developing to the fifth level of automation, which is the fully autonomous vehicle in which the driver has no tasks at all regarding the vehicle motion, and as the following description for the five levels [2] [3] of automation [1] [2]:

1. **Level 0** (No driving automation): In this level, the driver performs all aspects of the driving task, such as driving conventional automobiles.
2. **Level 1** (Driver assistance): The automation system provides sustained assistance with either steering or acceleration/braking, and an example of the system is adaptive cruise control [3] and lane-keeping assistance [4]. At this level, the automation system consists mainly of a controller such as a feed forward controller or a Proportional-Integral-Derivative (PID) controller [5], which could be designed using for example the Root Locus method [6], and it is most useful for a car operating on a slope by controlling the speed or preventing confusion by keeping the lane during a road turn.

*Not affiliated at the time of research

[2] https://www.sae.org/standards/content/j3016_202104/

[3] https://medium.com/@samiratra95/autonomous-driving-modular-pipeline-vs-end-to-end-and-llms-642ca7f4ef89

3. **Level 2** (Partial driving automation): An upgrade for the system at level 1 by providing both types of sustained assistance, steering and acceleration/braking. Examples of automation systems at this level are the Tesla Autopilot [4], Cadillac Super Cruise [5], and Ford BlueCruise [6]. We do not know how the companies are building the system, but from experience in systems and control, the goal can be achieved by implementing a controller for each task and keeping both operational at the same time.
4. **Level 3** (Conditional driving automation): A system at this level has a big difference from the previous level, where the system performs all aspects of the driving tasks within its Operational Design Domain (ODD) [7] [7] [8], with driver intervention expected upon request. Examples of systems at this level are the Audi Traffic Jam Pilot [8] and the Mercedes-Benz Drive Pilot [9]. The system, in this case, has a more complex structure, starting from sensors (LiDARs, Ultra Sonic), in addition to algorithms to process the sensor input and output control signals from the vehicle controllers. The link in the footnote shows a video of a demo for the Audi Traffic Jam Pilot [10]
5. **Level 4** (High Driving Automation): The system at this level performs all aspects of the driving task within its ODD, without expecting driver intervention. Some examples of these systems are Waymo's Robotaxis [11] and NAVYA's Shuttles [12]. At this point, the system is intelligent because it is in one or more parts using AI for processing and extracting information from cameras, LiDARs and other sensors or to predict the best trajectory for the vehicle, and the main difference between this system from a system of level 5 is that it is for a specific ODD and not built for all possible roads and driving scenarios.
6. **Level 5** (Full driving automation): At this level, the system is built to deliver an improved performance compared to a human driver, where the system performs all aspects of the driving task under all conditions without expecting driver intervention. An example of this is a concept vehicle without a steering wheel or pedals, like the Tesla Cybercab [13].

and as shown in the table in appendix A [1], a more detailed view of the automated vehicle functions in each level.

A lot of technologies has been built to improve the performance of the land vehicle autonomous driving systems, and there are mainly three approaches [9] [10] [11]:

- Modular pipeline.
- End-to-end.
- Large language Models and Multi-modal language models.

Each of the approaches listed addresses a group of the issues facing the one before it and introduces a group of possible improvements. The following sections will address the challenges of each approach and the solution suggested by the one after it, with a focus on the last approach and the improvements it shows, then provide summaries for a group of research papers that suggest large language models, vision language models, and multi-modal language models as solutions for the problems by replacing the full autonomous driving system or part of it while redesigning the rest of the system.

The following section of this research are Section 2, Modular pipeline and end-to-end, Section 3, MP and E2E challenges and LLM solutions, Section 4, Example systems, Section 5 Analysis, and Section 6, Conclusion.

After large models, the modularized end-to-end systems have delivered the best results in autonomous driving, and as will be clarified in the examples part of the research that the large models are some times improving a module of the system or substituting it, in other cases combining two or three modules but not always substituting and delivering the full system functionality.

[4] https://www.tesla.com/support/autopilot

[5] https://www.cadillac.com/technology/super-cruise?srsltid=AfmBOoreWh2WLT4WNRX0EYAvVE2Z9cSAsY3-vzOghbYBXf31VaCFt8

[6] https://www.ford.com/technology/bluecruise/

[7] https://www.sae.org/standards/content/j3259/

[8] https://magazine.audi.com.au/article/audi-ai-traffic-jam-pilot

[9] https://www.mercedes-benz.nl/passengercars/technology/drive-pilot.html?srsltid=AfmBOoo8iNXoLpZ8xQRonozBIw8w1Nh2PKpWh5noIshDEtztlwci9Ip_

[10] https://www.audi-mediacenter.com/en/videos/video/footage-audi-a8-audi-ai-traffic-jam-pilot-3785?source=post_page-----642ca7f4ef89----------

[11] https://waymo.com/

[12] https://www.navya.tech/en/solutions/moving-people/self-driving-shuttle-for-passenger-transportation/

[13] https://youtu.be/Qfj4urMF8CU

2 Modular pipeline and end-to-end

In this section will review the two classical approaches, the modular pipeline and the end-to-end, where in the later sections, the use and implementation of large language and vision models will be discussed on the basis of these two approaches.

2.1 Modular pipeline

This is the more traditional approach for creating autonomous driving systems, and it is not preferred today, mainly because of:

- Its complexity of design.
- The separation of each part of the system into different software and hardware.

which results in big challenges in optimizing the system and synchronizing every part of it. Though understanding it can give a great advantage when learning about more recent and popular approaches nowadays, since many approaches upgrade one module of the system, the modular pipeline approach breaks down the task of the autonomous driving system into four major parts (modules), and these are as follows.

2.1.1 Perception module

At this module, the vehicle collects, processes, comprehends and interprets the vehicle's surrounding environmental information by leveraging onboard sensors such as LiDAR, camera, radar, and event-based camera to extract as much information as possible about the road and surrounding objects such as other vehicles. Perception is a process that produces large amounts of data in the form of images, videos, point clouds and other formats [12]. The tasks and technologies of the perception module are mainly part of a computer vision system, such as 2D and 3D object detection, segmentation, object tracking, and sensor fusion [9]. For these tasks, algorithms like YOLO [13], Faster R-CNN [14], and VoxelNet [15] are implemented. In [16], the tasks of the perception module is defined in more detail, by introducing three categories:

- Object detection and identification.
- Depth estimation.
- Simultaneous location and mapping (SLAM).

where both the depth estimation and SLAM have three sensor options, the monocular, stereo and RGB-D.

For example [16], a Tesla vehicle uses:

1. A Wide-angle camera has a view angle of about 150 degrees responsible for recognising objects.
2. The medium-focal length camera has a view angle of 50 degrees, responsible for recognizing lane lines and close-by objects.
3. Telephoto cameras (Long-focus) have a view angle of 35 degrees and recognition distance of 200-250m

Some of the types of the features extracted from the images are categorized in the list:

- Edge features: Canny operator [17], Prewitt operator [18], Sobel operator [19], Laplacian operator [20], and Roberts operator.
- Appearance features: edges, contours, textures, dispersion, and topological characteristics.
- Statistical features: mean, variance, energy, entropy, autocorrelation coefficient and covariance.
- Transformation Coefficient Features: Fourier transformation, Hough transformation, Wavelet transformation, Gabor transformation, Hadamard transformation, and K-L transformation.
- Other features: the color type (greyscale, RGB) and color intensity.

Then the information and features from this module are passed to the Prediction module.

2.1.2 Prediction module

Built upon the real-time information received from the perception module, to function in analysing the past and current trajectories of the road users, such as pedestrians and other vehicles, then trying to predict the future short-term and long-term trajectories and behaviors of the road users to understand the future road conditions and ensure safety and stability [10]. Models for prediction tasks have four types according to [21]listed here:

- Feature encoding: where the trajectory is taken as sequential data, and many models are suggested for this case, such as RNN, Transformers [22], VectorNet [23] and MTP [24].
- Interaction modeling: where a module for the interaction between different vehicles, pedestrians and road elements is modeled, using transformers [25].
- Prediction head: where the probability of each trajectory is calculated by one of the previous two types, some of the approaches are using RNN [26] or take the prediction as a regression process and use MLP [27], besides other approaches.
- Generative model: such as [28] and [29].

2.1.3 Planning module

This module is responsible for generating the best path and trajectory from the current location to the target destination, based on the vehicle state (Position, speed, acceleration, direction) and environmental information received from the prediction and perception modules [9]. It has two types:

- **Global planning:** which is concerned with finding the best route from the starting point to the destination on the map using algorithms like A* and Dijkstra.
- **Local planning:** which involves real-time adjustments in speed and direction based on the current vehicle situation concerning the surroundings, common methods are RRT and deep learning-based planners.

In [30], the decision planning process was broken down into four layers. In this subsection will mention three of them, since according to the categorization of this paper, the fourth belongs to the next module, which is the control module, These three layers are:

- **Route planning:** Is done by finding the shortest path on the map from the current position to the destination.
- **Behavioral layer:** Makes instantaneous decisions about the way the vehicle should interact with the surrounding environment, and predicts the intentions and the behavior of the other objects around the vehicle. And that is done by machine learning.
- **Motion planning:** Based on the behavior decided a trajectory needs to be found for the vehicle. Using, for example, graph search methods.

And the fourth layer is the local feedback control layer, where the controller generates the actuator signal to deliver the trajectory.

2.1.4 Control module

This part of the system takes charge of executing the trajectory and the path received from the planning module, and it takes into account the vehicle dynamics model and environmental conditions to generate a range of control signals, such as acceleration/braking and steering. The more popular controller used at this step is the Model Predictive Controller (MPC) [31] [32].

2.2 End-to-end

A driving system that interprets the autonomous driving task as a singular learning task, by integrating separate components into a unified system, where it takes raw sensor signals as input and produces a plan as output [9]. This type of autonomous driving system provides more safety and reliability compared to the previous one. Besides other advantages [33] for end-to-end:

1. **Simplicity:** the system is simpler than the modular pipeline because it combines perception, prediction and planning in a single model that can be jointly trained.
2. **Optimization:** the possibility to optimize the system towards the ultimate task as a whole, including its intermediate representations.

3. **Shared backbones:** which increases computational efficiency.
4. **Data-driven optimization:** which improves the system by scaling the training resources.

The most popular methods for end-to-end systems are:

2.2.1 Imitation Learning

This refers to learning by demonstration and works by training the end-to-end algorithm to learn the functionality required by imitating the behavior of an expert algorithm [33] or a human driver. Even though it might be similar in concept, it is still so different from knowledge distillation in large language models [34] in application. This process requires a dataset containing trajectories collected from the expert algorithm while running, where each trajectory consists of a state-action pair, and the goal is for the end-to-end algorithm to learn to produce the same trajectories as the expert. Imitation learning has two types:

1. **Behavior cloning:** [35] This works by minimizing the loss as supervised learning over the dataset [36], the most commonly used method is the deep learning neural networks, such as the model suggested in [37], which is built of a convolutional neural network that can learn to steer a commercial vehicle on highways and residential roads. This approach has two shortcomings: Covariant shift and causal confusion, which can be tackled using techniques such as data augmentation [38] and data diversification [39] [40].
2. **Inverse Optimal Control (IOC):** [41] Also referred to as inverse reinforcement learning (IRL), in this approach the IOC algorithm learns an unknown reward function from experts demonstrations [33], for example a human is driving a vehicle and trying to optimize some unknown function with their actions, and the algorithm will be learning to behave similarly given the same state or situation on the road by watching the human driver. Approaches of this type include maximum entropy inverse reinforcement learning.

2.2.2 Reinforcement learning

This approach is based on trial-and-error, so the model usually gets built in a simulation environment such as CARLA [42] and Waymax [43] and then implemented to a real-size vehicle, while in case of building the model on a real-life vehicle a driver need to be present to take over as a safety measure. This approach requires significantly more data than imitation learning and training time.

Besides that, one approach to reduce the time required by the reinforcement learning algorithms while delivering similar results, the RL algorithm can be trained with imitation learning and then fine-tune it with reinforcement learning [44] [33].

Some of the algorithms used in this type are the Actor-Critic network and Deep Deterministic Policy Gradients. From [45] can find the steps of building a reinforcement learning system clearly defined, and as listed:

1. State space representation definition: which includes selecting the relevant sensor inputs and vehicle dynamics.
2. Designing the reinforcement learning architecture: which will make decisions for the vehicle motion, such as Deep Q-Network, and Proximal Policy Optimization (PPO).
3. Reward function design: which guides the reinforcement learning process to follow the safety guidelines and the traffic rules.
4. Uncertainty estimation and risk awareness: to enable the model to make risk-aware decisions and account for sensor noise and other errors, using algorithms such as Bayesian neural networks and Monte Carlo Dropout.
5. Simulation and real-world experiments: where the model gets trained on realistic scenarios in a simulation environment, for it to then be implemented in a real-world scenario.

In the next section will discuss some of the shortcomings of the two previously mentioned approaches and suggest another that will work as a solution, and improve the autonomous driving system.

3 Modular pipeline and end-to-end challenges and LLM solutions

Some of the challenges that faced the two previous approaches are generalization, interpretability, causal confusion, and robustness [14] [9]. Many research works have addressed these challenges and provided improvements for the autonomous driving system capabilities by adding an LLM or a VLM to the system [9] [33] [46] [47] [48]

[14] https://open.substack.com/pub/samerattrah/p/autonomous-driving-with-llms-vlms?r=2nuo7w&utm_campaign=post&utm_medium=web&showWelcomeOnShare=false

3.1 Challenges

1. The modularized pipeline approach challenges *only*:
    - Difficulty in integrating the different parts of the system, which mainly consists of four separate parts.
    - Unavoidable gaps between the system modules, in terms of time delays and accuracy of results, which result in bigger problems and a lack of performance going deeper into the system.
2. End-to-end approach challenges *only* [33]:
    - The information channels from one end of the system to the other are very long, which results in a waste of some information and delay.
    - The network structure is so complex, and that makes it difficult to debug or optimize the system.
    - The information and driving performance of the networks lack the human driver's common sense.
3. Explainability: where the results, actions and states of the algorithm in both approaches are unpredictable and understandable by humans in many scenarios [44].
4. Human-vehicle interaction: The human experience of interacting with the vehicle and the stability of the self-driving algorithm are lacking a lot of possible improvements and can be greatly improved.
5. Data scarcity: The recording of road data and creating datasets for autonomous vehicles with the correct type of annotation has not always been an easy process, which makes training models with higher performance more challenging.

3.2 Solutions

Some of the challenges of the modular pipeline have been solved by the end-to-end approach a long time ago, such as the separation and difficult synchronization between different parts of the system, where an end-to-end system solves the autonomous driving task as a single deliverable.

While the delay and accuracy of the system are challenges that still can be improved in all approaches, end-to-end also delivered a higher performance than the modular pipeline in overall results.

Some of the solutions that were provided by LLMs, VLMs and MLLMs to the challenges that were facing the end-to-end approach, in addition to the modular pipeline, are as follows: [46]

1. Perception: by demonstrating great performance and delivering tasks such as object referring and tracking, and open-vocabulary traffic environment perception, besides having improved performance in perception when data is scarce by relying on their few-shot learning characteristics [49] [50] [51].
2. Planning: In this field, the large models have shown great performance in open-world reasoning, and there are two types of large models [52] [53] [54]:
    - Fine-tuning pre-trained models: where there are many approaches and datasets to use for fine-tuning a large model to improve its performance in autonomous driving, some of the approaches are:
        (a) Parameter-efficient fine-tuning, such as low-rank adaptation [55] [56] [57].
        (b) Prompt tuning. such as learnable continuous vectors or soft prompts [58].
        (c) Instruction tuning [59] [49].
        (d) Reinforcement learning from human feedback [60].
    - Prompt engineering: Some methods tried to use the full reasoning potential with carefully engineered prompts, for example, using the chain-of-thought method to guide the reasoning process of the large model and get the best possible and accurate result [61].
3. Question answering: This use case improves the explainability of the model actions and provides a better human-vehicle-interaction, and understandability for the user, which could result in improved dataset generation and the possibility of optimizing the model [62] [63] [51] [64].
4. Generation: large models can use their generative ability to create videos for driving scenarios, and caption them, besides generating question-answer pairs or descriptions of driving scenes [65] [66] [67].

4 Example systems

In this section, we will mention a few examples for autonomous driving systems and large models implementations, and discuss there most significant strengths and differences in the architecture from the base model used in building the system and each one as the details mentioned in the research paper.

The systems will be categorized according to the improved functions they perform, and as the solutions list above shows, in addition to a few examples of end-to-end systems.

4.1 Perception

This subsection will mention the models, with special focus on their perception parts and architectures.

- **BEVFusion** [68]: This research presents a new sensor fusion approach, delivering improved results compared to the previous ones. Sensor fusion is the process of concatenating the features generated by several types of sensors, such as camera, LiDAR, and radar, into a single format, for it to be easily processed by the system parts that come after the perception module. Traditional sensor fusion approaches are:
  - Point-level fusion: this is an object-centric approach that usually paints image semantic features onto foreground LiDAR points and performs LiDAR-based detection on the decorated point cloud inputs.
  - Proposal-level fusion: which is also object-centric and works by creating object-proposals and projecting them to extract Regions of Interest (RoIs), one shortcoming of this approach is that it cannot trivially generalize to other tasks.

  BEVFusion delivers a big improvement on these approaches, and it works as follows:

  1. Apply modality-specific encoders to extract features.
  2. Transform multi-modal features to a unified BEV presentation that preserves both the geometric and semantic information.
  3. Apply pre-computation, which associates each point in the camera feature point cloud with a BEV grid.
  4. Interval reduction, which aggregates the features within each BEV grid by some symmetric function. and apply a specialized GPU kernel to accelerate the aggregation process.
  5. Apply a convolution-based BEV encoder to the unified BEV features.
  6. Append a few task-specific heads to support different 3D tasks.

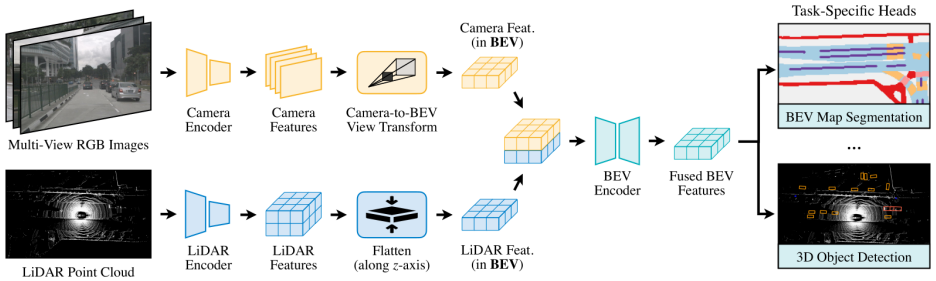

Figure 1: BEVFusion overview

As in the Figure 1, the network is built by using Swin transformer [69] for images and VoxelNet [15] for LiDAR, and applies a Feature Pyramid Network FPN to fuse them. Then, apply grid sampling with bilinear interpolation before each task-specific head to meet the output needs of any application task.

- **EMMA** [49]: Presenting an End-to-end multimodal model for autonomous driving. In which the model directly maps the camera sensor data into outputs, including planner trajectories, detected objects, and road graph elements. The model is built on top of Gemini [70], treating it as a first-class citizen of the system. The inputs to the model are:
  - Surrounded-view camera videos, to be processed for the perception tasks:
    1. 3D Object detection.
    2. Road graph estimation.
    3. Scene understanding.
  - High-level intent command.
  - A set of historical ego status.

  The model is trained using the chain-of-thought method, which improves the meta-decisions and critical object identification performance. The main functionalities of EMMA:

  - Generalizability.

- Predictive driving.
- Obstacle avoidance.
- Adaptive behavior.
- Accurate 3D detection.
- Reliable road graph estimation.

- **CarLLaVA** [51]: This paper presents a novel system which has a unique architecture from perception to control. The system uses the vision encoder from the LLaVA-NeXT [71] model, pre-trained on internet-scale vision data. for high-resolution images.

  The model takes camera images, the next two target points and the vehicle speed as input and gives time-conditioned waypoints with a PID controller for longitudinal control and space-conditioned waypoints with a PID controller for lateral motion as output.

  In the next stage, using a VLM, concatenate the features resulting from the vision encoder, then, using an adapter, downsample the feature map, and apply linear projection before generating the embeddings for the features.

  The system uses the LLaMA [72] architecture as a decoder, where mean squared error loss is used to monitor the waypoint generation optimization at training. and as in Figure 2

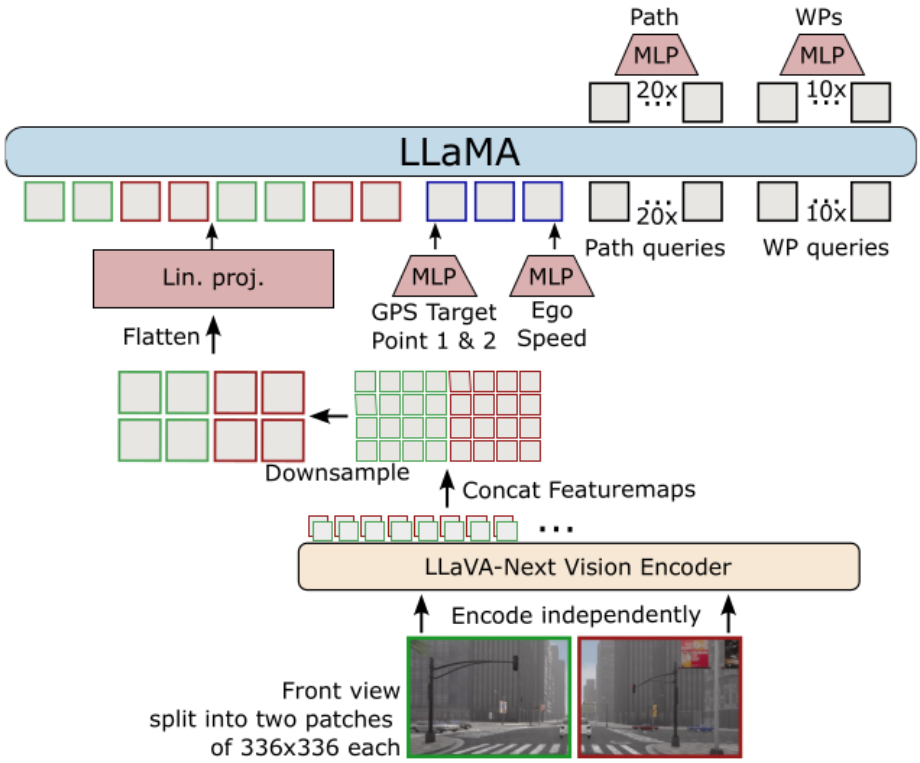

Figure 2: CarLLaVA architecture

The system includes the following properties and advantages:

- Camera only, without expensive labels such as BEV or depth.
- Vision language pre-training, which delivers an improvement on training from scratch.
- High resolution input images, to improve driving quality.
- Reduced training time.
- Semi-disentangled output representation.

This model is using the LLM and VLM as an end-to-end system to take images as input and produce trajectories as output, and needs to be connected to a PID controller. Besides that, this system is built to run on the CARLA simulator [42], and can be upgraded slightly to work on real-life vehicle driving.

4.2 Planning

This subsection will discuss the research papers that present the models built to deliver planning trajectories as output, focusing on what makes each one of them different in architecture and functionalities.

- **GPTDriver** [52]: This paper presents a model that represents the planner's inputs and outputs as language tokens instead of taking images and outputting trajectories like other systems, and it leverages the GPT-3.5 [73] [74] [75] LLM API to generate driving trajectories through a language description of coordinate positions. In addition to the model, it defines three motion planning approaches:
  - Rule-based: uses pre-defined rules to determine future driving trajectories.
  - Optimisation-based: formulate control planning as an optimal control problem.
  - Learning based: handle complex driving scenarios by learning from large-scale human driving data, which is the one used in this system implementation.

  One of the improvement aspects of this model implementation is the high interpretability, where it uses a chain-of-thought [76] reasoning strategy, which works in three steps:

  1. The planner identifies the critical objects from the perception results.
  2. By analyzing the future motions of these critical objects, the planner should infer when, where, and how this critical object may influence the ego vehicle.
  3. Using the insights gained from the previous analyses, the planner draws a high-level driving decision and converts it into a planned trajectory.

  Another main difference for this model from the others is that it uses the model API, instead of using the model implementation, and editing the architecture to build a complete autonomous driving system.
- **MTD-GPT** [53]: The model is built to manage multiple driving tasks specifically in unsignalized intersections. It introduces a pipeline to build a model by (1) training a group of single-task reinforcement learning expert models using PPO algorithm [77], (2) sampling data from the expert model's performance in the environment, and (3) utilising the mixed multi-task dataset created from stacking the expert samples to train a GPT-2 [78] model offline, and the final step (4) GPT evaluation.
- **VLM-AD** [54]: This research suggests integrating a VLM, where the one used is GPT-4o [75], into a modular pipeline system in a knowledge distillation approach [34] to improve its planning module performance, where it uses the VLM as a teacher to provide extra supervision, that will enhance the features and therefore the trajectory produced by the planning module.

  The VLM is used for training only, not inference, and that is because using the LLM or VLM directly to reason is time and resource-consuming since the language representation is difficult to transfer into a control command or a planned trajectory, as in Figure 3 (b)

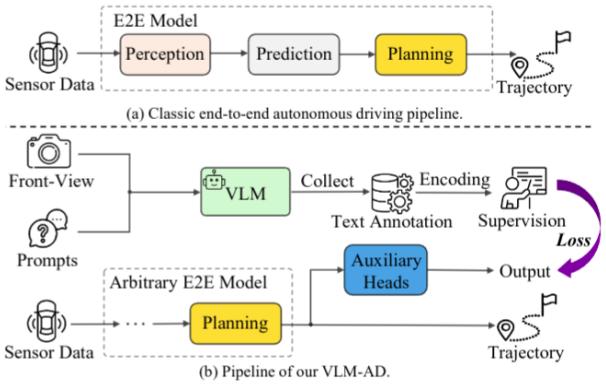

Figure 3: VLM-AD architecture [54]

The process of training this system starts when the VLM model receives the images from around the vehicle and provides an annotation for the state of the vehicle, then the VLM model gets prompted with questions to further explain the state and decision and provide further information and reasoning to enrich the understanding of the traditional trajectory model.

In the next step, the auxiliary heads shown in the Figure 3 receive the ego information from the traditional trajectory model, and align it with the language instructions from the VLM to get an improved decision. and the network used for the heads is a language model such as CLIP [79].

- **LanguageMPC** [80]: This paper implements the LLM [73] [74] [75] to function as the brain of the AD system. Which takes a prompt and the environment perception information, and uses them to make the high-level decisions and creates a low-level mathematical representation to be input into a Model-Predictive-Controller (MPC) [81]. In addition to planning for vehicle driving, it presents a chain-of-thought framework [76] for the driver to have a conversation with the LLM to improve interoperability of the driving decisions.

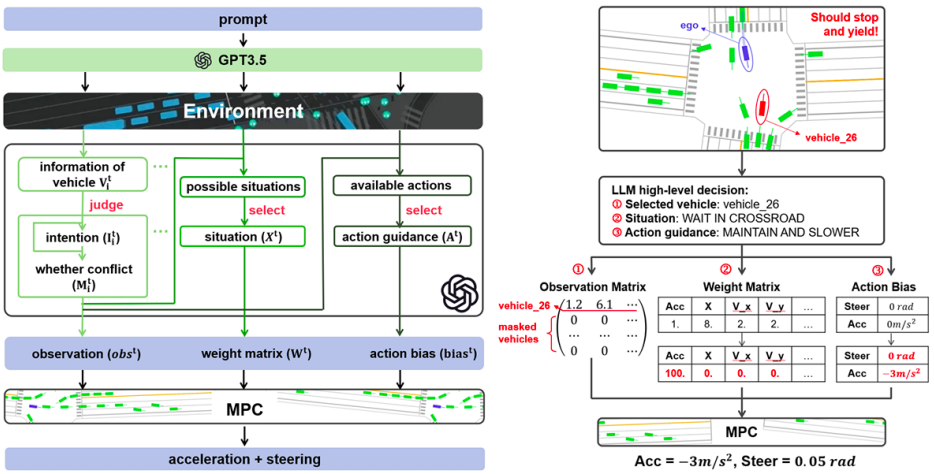

Figure 4: LaguageMPC architecture [80]

The model functions as follows: the LLM gather information, reasons, and renders judgments. Then, from left to right branches in the center of Figure 4 (a), the LLM will:

  - Identify the vehicles requiring attention in the surrounding environment.
  - Evaluate the situation.
  - Offer action guidance for the ego vehicle.

  And the results from all the processes, the LLM gives attention to each vehicle separately and determines if they pose a conflict or not. And from the attention it creates the observation matrix. The weight matrix defines a situation (acceleration and steering) that a vehicle could be in, while the bias, as shown as the action bias matrix, is the value of the steering or acceleration as scalar values.

- **OccLLaMA** [82]: This research paper presents a unified multi-modal vocabulary, for vision, language and action (VLA), to unify the VLA-related tasks, including but not limited to scene understanding, planning and 4D occupancy forecasting. using a model based on LLaMA [72] [83] and a VQVAE-like [84] architecture as a tokenizer while using occupancy mapping as perception.

  At training time, after a few steps of processing to get the point cloud features and aggregate them using a pillar embedding, and employ a Swin Transformer block [69] to obtain the BEV feature map. It uses vector quantization to obtain discrete representations. After quantization, the decoder restores 3D voxel features by stacking a convolution block and an upsampling layer.

  For the training process, it utilizes three loss functions similar to [85]:

  - Cross-entropy for geometry.
  - Lovasz-softmax for semantics.
  - Embedding loss for codebook learning.

  This system uses the LLM as an end-to-end solution to the autonomous driving problem.

- **DriveVLM** [86]: This research paper presents a model plan for the vehicle motion and provide explanation by the chain-of-thought process, which has three modules:

  - Scene description: Linguistically describe the environment and critical objects.
  - Scene analysis: describes the critical objects and their influence on the ego vehicle.

– Hierarchical planning: formulates plans from meta-actions to waypoints.

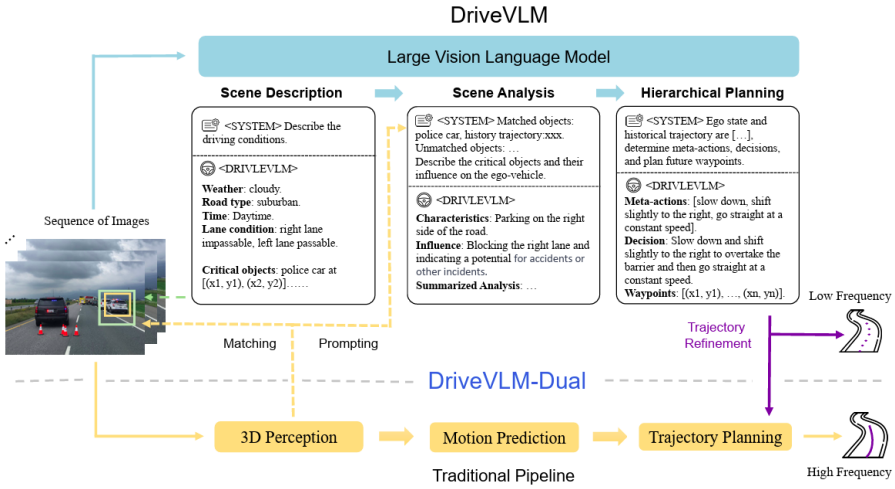

Figure 5: DriveVLM structure.

As shown in the Figure 5, the model has two parts: DriveVLM and DriveVLM-Dual, where the latter incorporates, in addition to the VLM functions, a 3D-perception module and trajectory planning refinement.

The VLM used is Qwen-VL [87] consists of a vision transformer encoder, an attention-based extractor and an LLM.

The scene description module identifies critical objects, which are analyzed in three stages:

– Static attributes: describe inherent properties, such as a truck's cargo which could cause a hazard.
– Motion states: describe object dynamics, such as position and speed.
– Particular behaviors: describe special actions.

The scene analysis dives into the details of the objects, then the hierarchical planning module generates the plan.

- **DriveMM** [88]: The model takes images and videos in addition to a user instruction as input, and gets pretrained using the curriculum learning [89] method. to deliver outputs in three tasks: perception, prediction, and planning.

  It adapts LLaVA [90], which has three parts:

  – Vision encoder, such as SigLIP [91].
  – Projector.
  – LLM, such as LLaMA 3.1 [92].

  The training process of the system is divided into four steps:

  1. Language-image alignment.
  2. Single-image pre-training.
  3. Multi-capacity pre-training.
  4. Driving fine-tuning.

4.3 Explainability

This section will discuss the autonomous driving systems with attention to the details of the explainability of the applications and architecture of the models used.

- **Agent Driver** [62]: Discusses an LLM-based autonomous driving system [52], which introduces:
  – A versatile tool library that has four modules:
    * Detection [93].
    * Prediction [94].
    * Occupancy [95].

    - Mapping [96].
  - A cognitive memory of common sense contains two sub-memories:
    - Commonsense memory.
    - Experience memory.
  - A reasoning engine capable of chain-of-thought reasoning consists of four core components:
    - Chain-of-thought reasoning: it helps learning by generating a text output for each object in the environment.
    - Task planning: produce high-level driving planning.
    - Motion planning: generates a text-based trajectory.
    - Self-reflection: it is a collision check and optimization approach.

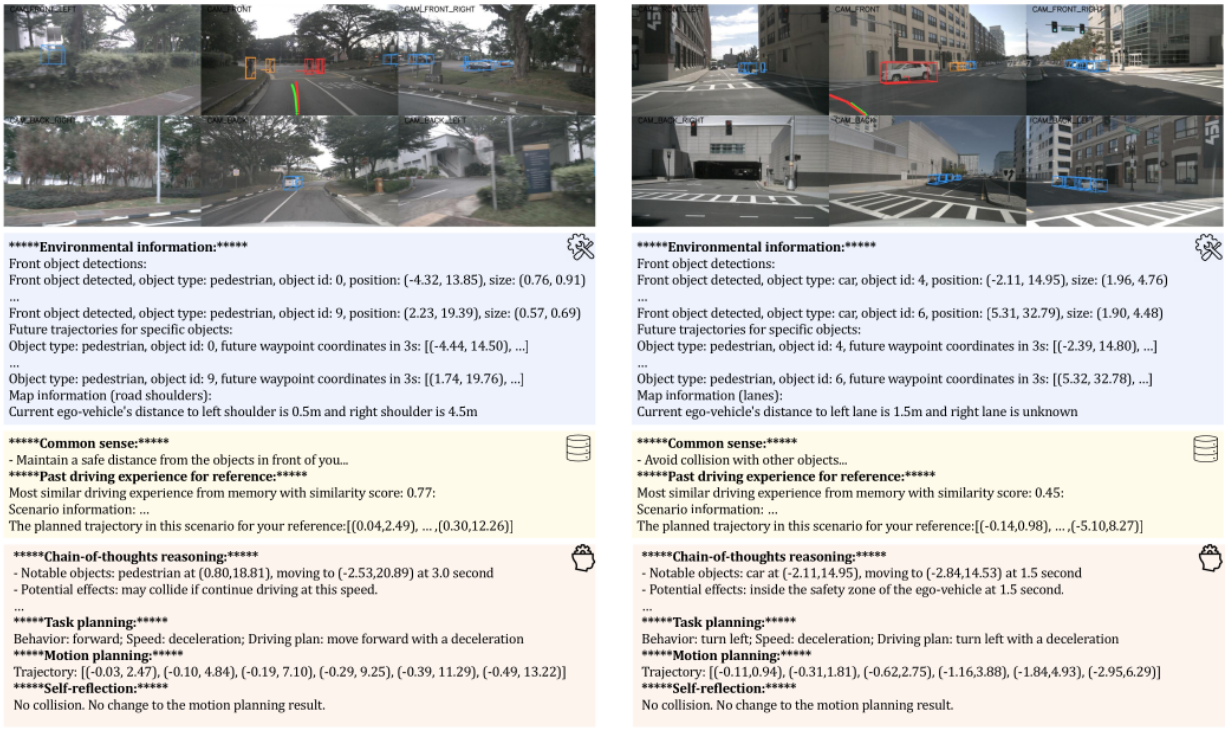

Figure 6: Interpretability of Agent-Driver [62]

These three components are coordinated by LLMs. The driving process starts when the system takes sensory data, process them with neural networks, then the output from the neural networks using a *tool library* get processed by and LLM to generate text messages for each driving scenario, then the LLM output get used as query for the *cognitive memory* to retrieve any relevant traffic rules or previous experiences, after that a *reasoning engine* takes the perception and traffic rules as input, and generate a trajectory.

As in Figure 6, the output messages of LLMs from the tool library, cognitive memory, and reasoning engine are recorded during system execution. And because of this, the whole driving decision-making process is interpretable and explainable.

- **DriveMLM** [63]: The system suggested in this research paper consists of a model built to run on simulators such as CARLA [42]. And to be easily integrated with modular AD systems, it standardizes the motion states and introduces a motion (behavior) planning module, which takes input from (1) Driving rules, (2) User commands, and (3) Sensors such as a camera and a LiDAR.

  Most notable about this research is that it introduces a data engine to predict motion decisions, explain them, and generate annotated datasets.

  The topology of the framework has three parts:

  - Behavioral planning states alignment: aligns an LLM with a pre-trained behavior planning module. to be able to use the LLM as a behavior planner.

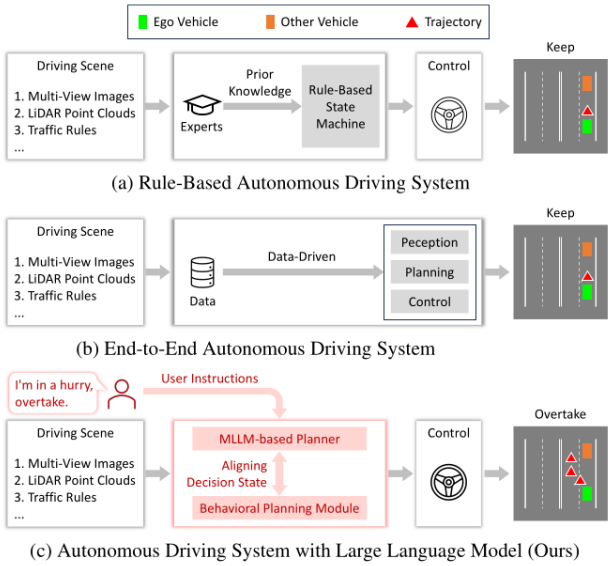

Figure 7: Comparison of the DriveMLM system structure with classical approaches [63]

- MLLM planner: consists of a tokenizer to generate tokens from the input and a decoder to generate control signals from the tokens.
- Data collection strategy.

As shown in the Figure 7, the tokenizer for the images input used is the QFormer [97]. For LiDAR data, the Sparse pyramid transformer [98] is used as a feature extractor, which gets input to QFormer to get embeddings

- **LingoQA** [99]: The research introduces a benchmark, to test video question answering and a dataset in addition to a vision language model for autonomous driving.

  The model is trained to answer questions about driving scenes as well as general knowledge. The research paper defines an evaluation metric called Lingo Judge, in contrast to the GPT Judge [100]. which delivers a 0.950 Spearman and 0.993 Pearson correlation coefficient. and built on a learned text classifier.

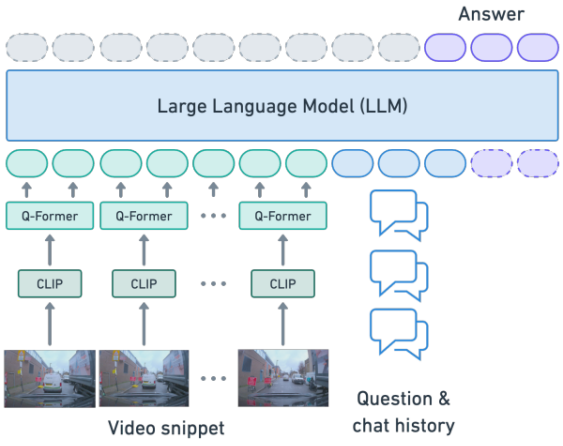

Figure 8: LingoQA architecture

As shown in Figure 8, the image encoder used is CLIP [79], and from that, the features are input into a Q-former initialized from Blip-2 [97], to transform the vision features into language features. and then get inputted into LLaMA-2 [72] model initialized on Vicuna v1.5 7B [101].

4.4 Data generation

One application for the large vision language models is generating videos and images for driving scenarios, which could be used to create new datasets for training autonomous driving systems or improve and expand existing datasets.

- **VQA-Diff** [66]: The research presents a model that improves the quality of the images generated by diffusion models by adding fine definitions to the vehicles and generated environment, such as the face features of the pedestrians, the car manufacturers, and the model of the vehicle running on the street. The system is built of a group of diffusion models and an LLM able to learn from an in-the-wild observation, with zero-shot learning prediction.

  The system has three parts:

  – VQA large language model text-to-text [97]: generates the necessary information about the view.
  – Multi-expert diffusion models [102]: generate a multi-view structures of the vehicle.
  – Edge-to-image ControlNet [103] [104]: renders the multi-view structures into photorealistic novel view.

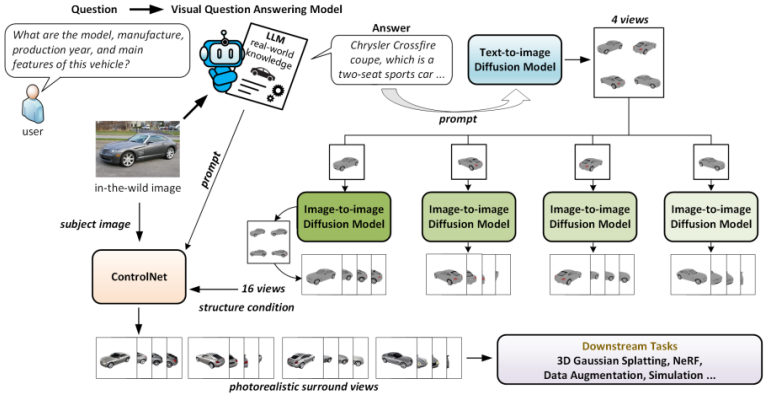

Figure 9: VQA-diff [66] framework

The system works as shown in Figure 9 by:

1. Inputting the LLM [97] response with the subject image into a text-to-image [105] diffusion model.
2. Using the output of the text-to-image diffusion, which is a single image with four scenes of the same vehicle, and it get split to four images with a quarter of the area of the first one, and inputted to four image-to-image diffusion models [102] to generate multi-view images from the same subject.
3. Inputting a prompt, subject image, and 16 views structure condition generated from the group of image-to-image diffusion models into ControlNet [103] [104].
4. Using the output of ControlNet, which is a photorealistic surround view in a downstream task.

- **GAIA-1** [65]: In this paper, a generative system is reported where the system has three inputs: video, text, and action, that get encoded into a shared space, as shown in Figure 10. GAIA-1 works by partitioning the model into two parts:

  – The world model [106]: reason about the scene components and dynamics, casts the world modeling as an unsupervised sequence modeling problem. using an autoregressive transformer network.
  – The video diffusion decoder [107]: translates the latent representations into high-quality representations. It is a 3D U-Net with factorized spatial and temporal attention layers.

  The model works as follows:

  1. Tokenize the three inputs each according to a specific criterion. where the image tokenizer has a trade-off between the vocabulary size and the sequence length. And for text, there are two choices: character or word level tokenization.
  2. Tokenizing the video input can be done using a discrete image autoencoder on two stages:
     (a) Compress the information from low pixels.
     (b) Guides the process towards meaningful representation.
  3. The discrete image autoencoder is a fully convolutional 2D U-Net [108], and the losses used to train it are:

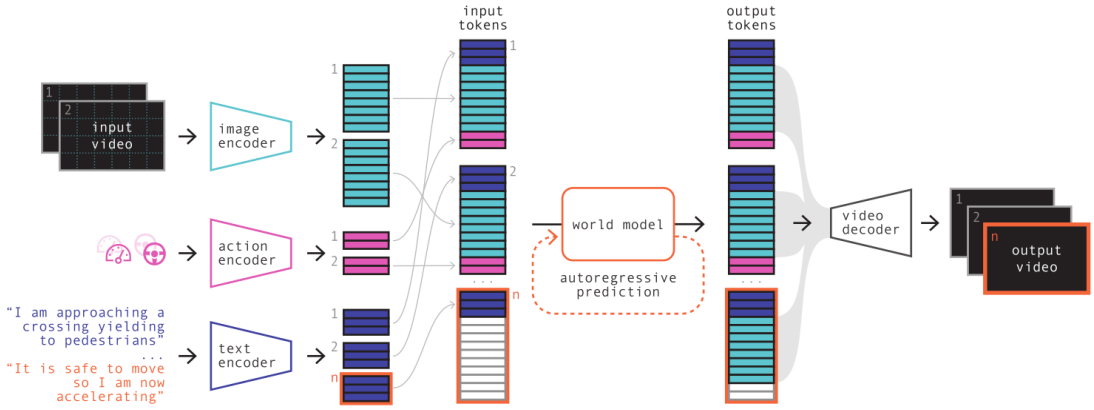

Figure 10: GAIA-1 architecture [65]

  - Image reconstruction loss.
  - Quantization loss.
  - Inductive bias loss.

- **GAIA-2** [109]: This research paper is based on GAIA-1 [65] and delivers improvements in performance, where this model gives improved control to the environment and the ego vehicle in comparison to the previous one. GAIA-1 used a discrete latent variable while GAIA-2 uses a continuous latent space.

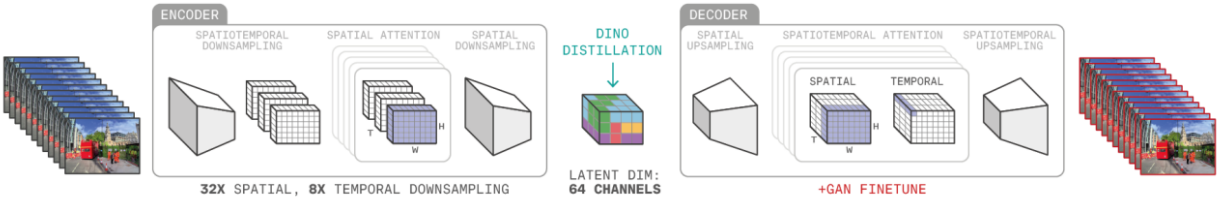

Figure 11: GAIA-2 video tokenizer [109]

The model has two main parts:

  - Video tokenizer as in Figure 11. The video encoder compresses the input videos into a compact continuous latent space for the world model to learn from them, predicting the future latent states. and it is made of a space-time factorized transformer, with an asymmetric encoder-decoder architecture.
  - Latent world model: as in the Figure 12, which is a time-space factorized transformer trained using flow matching. and takes the inputs as:
    * Past latents.
    * Actions.
    * Conditioning inputs: which includes dynamic agent properties, camera configurations.

- **SimBEV** [67]: This research paper presents a generative model and a dataset in the form of Bird's Eye View (BEV), and suggests that BEV perception is interesting for two main reasons:
  - Conductive to the fusion of information from different modalities.
  - BEV Segmentation offers a concise view of the environment.

  The model relies on the CARLA simulator [42] with a custom content library. At the time of generating the dataset, the model runs on CARLA to generate videos of a car's surrounding view, and offers two types of annotation:

  - 3D object bounding boxes.
  - BEV ground truth, for models like BEVFusion [68] to be trained on.

  It works by randomizing (within some bounds) as many simulation parameters as possible.

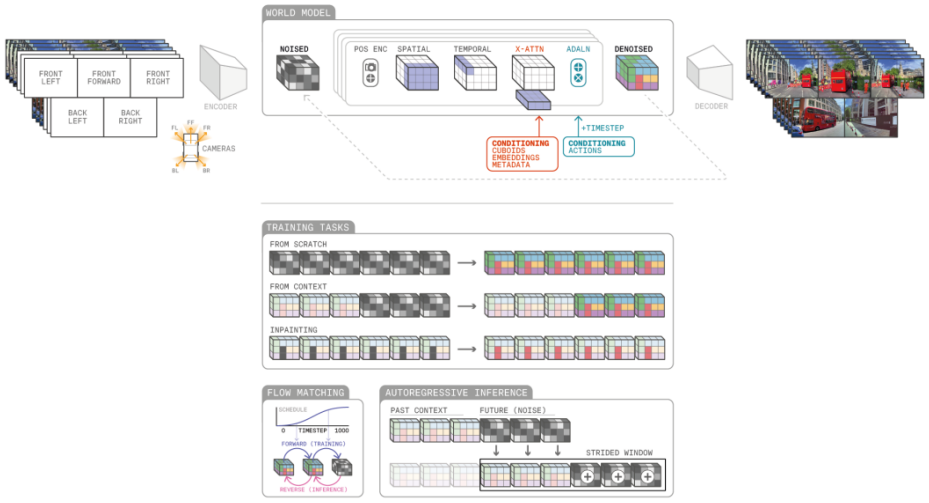

Figure 12: GAIA-2 world model

4.5 End-to-end

- **LMDrive** [50]: This model processes and integrates inputs from multi-modal sensor (camera - LiDAR) data with natural language instructions. And the output of the model is control signals for the vehicles.

  LMDrive consists of two parts:

  - Vision encoder generates visual tokens. The one used is designed differently from CLIP, which is the best way to align vision and language, and it works as follows:
    - Implements a ResNet [110] backbone to extract visual features and, after a few steps of processing, get the point clouds that will be aggregated into Bird's Eye View (BEV) queries.
    - BEV decoder that will process the visual features into tokens of three types:
      1. BEV tokens.
      2. Waypoint tokens.
      3. Traffic light tokens.
  - LLM, uses LLaMA [72], with its associated parts (tokenizer [72], Q-former [97], and adapters) that take the vision tokens and natural language instructions, then generate control commands from them. And as the Figure 13 shows:

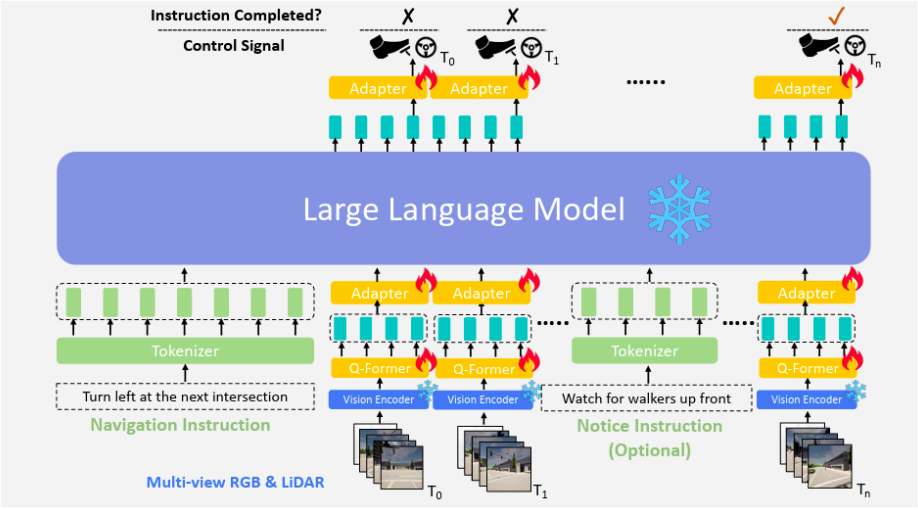

Figure 13: LMDrive architecture

The training process of the complete system has two stages:

1. Vision encoder pre-training stage. Training the encoder is in three parts:
   (a) Object detection.
   (b) Future waypoint prediction.
   (c) Traffic light status classification.
2. Instruction fine-tuning stage.

While the LLM weights stay frozen at all stages of the training and inference. From the model details and the Figure 13 can find that the model is implementing the LLM as an end-to-end system.

- **DriveGPT4** [111]: Presenting a multi-modal LLM based on LLaVA [90] capable of processing multi-frame video captured by a front-view camera as input and textual queries.
  The output of the model is the prediction of the control signal for the next step, besides a natural language answer to the text query.

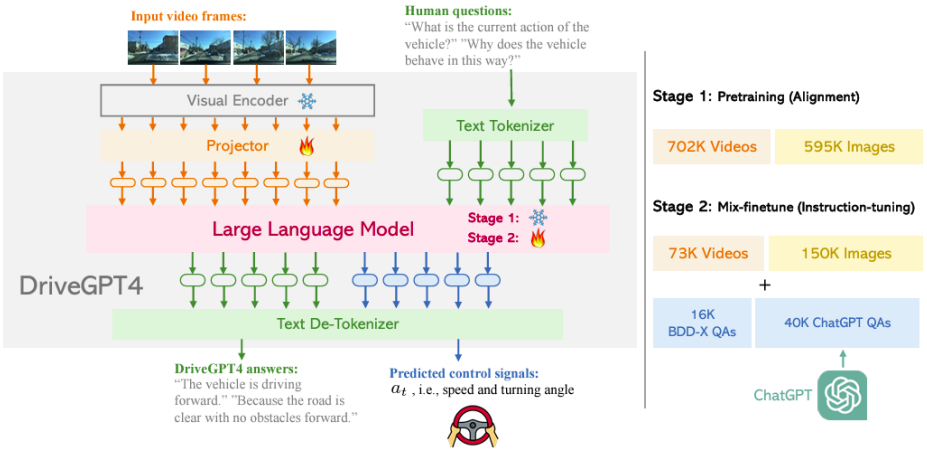

Figure 14: DriveGPT4 architecture

DriveGPT4 consists of:

- Video tokenizer: consists of a visual encoder (CLIP) [79] and a projector, converts video frames into text domain tokens.
- LLaMA-2 [72] as LLM.
- De-tokenizer inspired from RT-2 [112].

The training process for the model is in two stages:

- The pre-training stage focused on video-text alignment, using CC3M and WebVid-2M [113] datasets, where only the projector is trained and the encoder and LLM weights are fixed.
- The mix-finetuning stage aimed at training the model for question answering, and the part trained is the LLM alongside the projector.

- **RAG-Driver** [114]: This research presents a generalist model that delivers state-of-the-art performance and exhibits exceptional zero-shot performance by the usage of the Retrieval Augmented Generation (RAG) [115] method to improve the end-to-end autonomous driving system. The model addresses the following challenges:

  1. Explainability.
  2. Data scarcity.
  3. Expensive training requirements.
  4. Forgetting.

  The model delivers three outputs:

  - Action explanation.
  - Action justification.
  - Next control signal prediction.

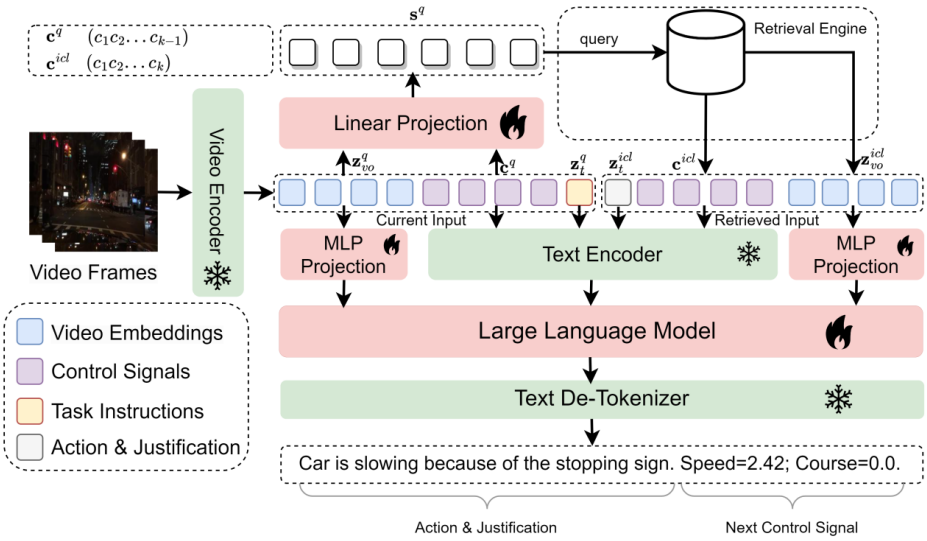

Figure 15: RAGDriver overview

RAG-driver has two main components that interact through a retrieval engine:

- Unified perception and planning unit, built on MLLM backbone, which is Vicuna 1.5 7B [101].
- Memory unit: built upon a hybrid vector and textual database and the retrieval engine.

5 Analysis

Considering the research papers discussed and the models being used in implementing an autonomous driving system, can find that the models of interest are mainly three:

1. LLaVA [71] [90].
2. LLaMA [72] [83] [92].
3. GPT [52] [73] [74] [75].

While there are others, such as Gemini [70] and Vicuna [101], they are not used often by researchers in this field.

Another notice is that the planning module is the most researched and suggested for improvement, but the largest change and the approach that might have the biggest potential for delivering an improved autonomous driving system is the end to end system that is mentioned in the examples LMDrive, DriveGPT4, and RAG-Driver, where they take the perception information as input and produce the control signals as output, and that is because the improvement of the large models is still an actively searched subject, to improve the capabilities of the models while keeping the same size or decreasing it.

In addition to the mentioned models and architectures, a group of the most popular and used datasets are in Table 1 [9] [44]

6 Conclusion

The systems that include large language models and large vision models can be a solution for many challenges for the previous approaches, and some of them can deliver improvements to the modular pipeline and end-to-end systems, in addition to possibly replacing them fully or partially.

The LLMs enable the building of a new type of system, which is more inclusive than end-to-end, where, in contrast to the traditional end-to-end, it does not need a controller after the last step but delivers the controller signal as output after taking the perception information as input.

Dataset	Task
KITTI [116]	3D & 2D Object Detection, Semantic Segmentation, Object Tracking
Cityscapes [117]	3D & 2D Object Detection, Semantic Segmentation
CityFlow [118]	Object Tracking, Re-Identification
nuScenes [119]	3D & 2D Object Detection, 3D & 2D Semantic Segmentation, Object Tracking, Motion Planning
BDD100K [120]	2D Object Detection, 2D Semantic Segmentation, Object Tracking
Waymo [121]	3D & 2D Object Detection, 2D Semantic Segmentation, Object Tracking
BDD-X [122]	2D Object Detection, 2D Semantic Segmentation, Object Tracking, Action Explanation
NuPrompt [123]	Multiple Object referring and tracking
Talk2BEV [124]	Visual question Answering,
DRAMA [125]	Image Captioning, Visual Question Answering
DeepDrive-X [122]	Global Navigation Satellite system, Text annotation

Table 1: Datasets and their supported tasks

In addition to that, the LLMs enable data generation, which is a task that was not addressed by the modular pipeline and the classical end-to-end, since video, images, and text datasets are scarce in usual, so generating and simulating driving environments would be a great addition to the autonomous driving industry.

Explainability is another application for large vision and language models that did not exist as part of the other two approaches, it improves the possibility of optimizing the systems further by understanding their behavior and communicating with some of them.

So we can notice that large language models and large vision models can transform the autonomous driving system into a more similar experience to a human driver.

References

[1] Eilat Navon Nathan. Challenging the silences: An analysis of sae j3016 as a classification system. 2024.

[2] Debbie Hopkins and Tim Schwanen. Talking about automated vehicles: What do levels of automation do? *Technology in Society*, 64:101488, 2021.

[3] Yangsheng Jiang, Hongwei Cong, Hongyu Chen, Yunxia Wu, and Zhihong Yao. Adaptive cruise control design for collision risk avoidance. *Physica A: Statistical Mechanics and its Applications*, 640:129724, 2024.

[4] Sijie Wei, Peter E. Pfeffer, and Johannes Edelmann. State of the art: Ongoing research in assessment methods for lane keeping assistance systems. *IEEE Transactions on Intelligent Vehicles*, 2023.

[5] Khaled Sailan, Klaus Dieter Kuhnert, et al. Modeling and design of cruise control system with feedforward for all terrian vehicles. *Computer Science & Information Technology (CS & IT)*, 1(2):339–349, 2013.

[6] M Manju Prasad and MA Inayathullah. Root locus approach in design of pid controller for cruise control application. In *Journal of Physics: Conference Series*, volume 2115, page 012023. IOP Publishing, 2021.

[7] Thor Myklebust, Tor Stålhane, and Dorthea Mathilde Kristin Vatn. Definition of the system, operational design domain, and concept of operation. In *The AI Act and The Agile Safety Plan*, pages 19–27. Springer, 2025.

[8] Marcel Aguirre Mehlhorn, Andreas Richter, and Yuri AW Shardt. Ruling the operational boundaries: A survey on operational design domains of autonomous driving systems. *IFAC-PapersOnLine*, 56(2):2202–2213, 2023.

[9] Xingcheng Zhou, Mingyu Liu, Ekim Yurtsever, Bare Luka Zagar, Walter Zimmer, Hu Cao, and Alois C Knoll. Vision language models in autonomous driving: A survey and outlook. *IEEE Transactions on Intelligent Vehicles*, 2024.

[10] Xu Wang, Mohammad Ali Maleki, Muhammad Waqar Azhar, and Pedro Trancoso. Moving forward: A review of autonomous driving software and hardware systems. *arXiv preprint arXiv:2411.10291*, 2024.

[11] Yuxuan Zhu, Shiyi Wang, Wenqing Zhong, Nianchen Shen, Yunqi Li, Siqi Wang, Zhiheng Li, Cathy Wu, Zhengbing He, and Li Li. Will large language models be a panacea to autonomous driving? *arXiv preprint arXiv:2409.14165*, 2024.

[12] Yun Li, Kai Katsumata, Ehsan Javanmardi, and Manabu Tsukada. Large language models for human-like autonomous driving: A survey. In *2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC)*, pages 439–446. IEEE, 2024.

[13] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[14] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

[15] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018.

[16] Fei Liu, Zihao Lu, and Xianke Lin. Vision-based environmental perception for autonomous driving. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 239(1):39–69, 2025.

[17] Geng Hao, Luo Min, and Hu Feng. Improved self-adaptive edge detection method based on canny. In *2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics*, volume 2, pages 527–530. IEEE, 2013.

[18] Weijie Zhou, Xiaoyu Du, and Senhao Wang. Techniques for image segmentation based on edge detection. In *2021 IEEE International Conference on Computer Science, Electronic Information Engineering and Intelligent Control Technology (CEI)*, pages 400–403. IEEE, 2021.

[19] Theodora Sanida, Argyrios Sideris, and Minas Dasygenis. A heterogeneous implementation of the sobel edge detection filter using opencl. In *2020 9th International conference on modern circuits and systems technologies (MOCAST)*, pages 1–4. IEEE, 2020.

[20] Liang Pei, Zhiwei Xie, and Jiguang Dai. Joint edge detector based on laplacian pyramid. In *2010 3rd International Congress on Image and Signal Processing*, volume 2, pages 978–982. IEEE, 2010.

[21] Jianbang Liu, Xinyu Mao, Yuqi Fang, Delong Zhu, and Max Q.-H. Meng. A survey on deep-learning approaches for vehicle trajectory prediction in autonomous driving. *2021 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 978–985, 2021.

[22] Jiacheng Pan, Hongyi Sun, Kecheng Xu, Yifei Jiang, Xiangquan Xiao, Jiangtao Hu, and Jinghao Miao. Lane-attention: Predicting vehicles' moving trajectories by learning their attention over lanes. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7949–7956, 2019.

[23] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11525–11533, 2020.

[24] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider, and Nemanja Djuric. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *2019 international conference on robotics and automation (icra)*, pages 2090–2096. IEEE, 2019.

[25] Yicheng Liu, Jinghuai Zhang, Liangji Fang, Qinhong Jiang, and Bolei Zhou. Multimodal motion prediction with stacked transformers. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7573–7582, 2021.

[26] Haoran Song, Wenchao Ding, Yuxuan Chen, Shaojie Shen, Michael Yu Wang, and Qifeng Chen. Pip: Planning-informed trajectory prediction for autonomous driving. In *European Conference on Computer Vision*, 2020.

[27] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Benjamin Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, Congcong Li, and Dragomir Anguelov. Tnt: Target-driven trajectory prediction. In *Conference on Robot Learning*, 2020.

[28] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher Bongsoo Choy, Philip H. S. Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2165–2174, 2017.

[29] Sriram Narayanan, Buyu Liu, F. Pittaluga, and Manmohan Chandraker. Smart: Simultaneous multi-agent recurrent trajectory prediction. In *European Conference on Computer Vision*, 2020.

[30] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1(1):33–55, 2016.

[31] Erfan Aasi, Mingyu Cai, Cristian Ioan Vasile, and Calin A. Belta. A two-level control algorithm for autonomous driving in urban environments. *IEEE Transactions on Intelligent Transportation Systems*, 26:410–424, 2025.

[32] Arun Balajee Vasudevan, Neehar Peri, Jeff Schneider, and Deva Ramanan. Planning with adaptive world models for autonomous driving. *arXiv preprint arXiv:2406.10714*, 2024.

[33] Li Chen, Penghao Wu, Kashyap Chitta, Bernhard Jaeger, Andreas Geiger, and Hongyang Li. End-to-end autonomous driving: Challenges and frontiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[34] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[35] Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, 1995.

[36] Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *Conference on robot learning*, pages 66–75. PMLR, 2020.

[37] Mariusz Bojarski, David W. del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *ArXiv*, abs/1604.07316, 2016.

[38] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6:1–48, 2019.

[39] Felipe Codevilla, Eder Santana, Antonio M. López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9328–9337, 2019.

[40] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv preprint arXiv:1812.03079*, 2018.

[41] Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2008.

[42] Alexey Dosovitskiy, Germán Ros, Felipe Codevilla, Antonio M. López, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on Robot Learning*, 2017.

[43] Cole Gulino, Justin Fu, Wenjie Luo, George Tucker, Eli Bronstein, Yiren Lu, Jean Harb, Xinlei Pan, Yan Wang, Xiangyu Chen, John D. Co-Reyes, Rishabh Agarwal, Rebecca Roelofs, Yao Lu, Nico Montali, Paul Mougin, Zoey Yang, Brandyn White, Aleksandra Faust, Rowan Thomas McAllister, Drago Anguelov, and Benjamin Sapp. Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research. *ArXiv*, abs/2310.08710, 2023.

[44] Ardi Tampuu, Tambet Matiisen, Maksym Semikin, Dmytro Fishman, and Naveed Muhammad. A survey of end-to-end driving: Architectures and training methods. *IEEE Transactions on Neural Networks and Learning Systems*, 33(4):1364–1384, 2020.

[45] Dantong Xiang. Reinforcement learning in autonomous driving. *Applied and Computational Engineering*, 2024.

[46] Zhenjie Yang, Xiaosong Jia, Hongyang Li, and Junchi Yan. Llm4drive: A survey of large language models for autonomous driving. *ArXiv*, abs/2311.01043, 2023.

[47] Daocheng Fu, Xin Li, Licheng Wen, Min Dou, Pinlong Cai, Botian Shi, and Y. Qiao. Drive like a human: Rethinking autonomous driving with large language models. *2024 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, pages 910–919, 2023.

[48] Bo Jiang, Shaoyu Chen, Bencheng Liao, Xingyu Zhang, Wei Yin, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Senna: Bridging large vision-language models and end-to-end autonomous driving. *ArXiv*, abs/2410.22313, 2024.

[49] Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, et al. Emma: End-to-end multimodal model for autonomous driving. *arXiv preprint arXiv:2410.23262*, 2024.

[50] Hao Shao, Yuxuan Hu, Letian Wang, Guanglu Song, Steven L Waslander, Yu Liu, and Hongsheng Li. Lmdrive: Closed-loop end-to-end driving with large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15120–15130, 2024.

[51] Katrin Renz, Long Chen, Ana-Maria Marcu, Jan Hünermann, Benoit Hanotte, Alice Karnsund, Jamie Shotton, Elahe Arani, and Oleg Sinavski. Carllava: Vision language models for camera-only closed-loop driving. *arXiv preprint arXiv:2406.10165*, 2024.

[52] Jiageng Mao, Yuxi Qian, Junjie Ye, Hang Zhao, and Yue Wang. Gpt-driver: Learning to drive with gpt. *arXiv preprint arXiv:2310.01415*, 2023.

[53] Jiaqi Liu, Peng Hang, Xiao Qi, Jianqiang Wang, and Jian Sun. Mtd-gpt: A multi-task decision-making gpt model for autonomous driving at unsignalized intersections. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 5154–5161. IEEE, 2023.

[54] Yi Xu, Yuxin Hu, Zaiwei Zhang, Gregory P Meyer, Siva Karthik Mustikovela, Siddhartha Srinivasa, Eric M Wolff, and Xin Huang. Vlm-ad: End-to-end autonomous driving through vision-language model supervision. *arXiv preprint arXiv:2412.14446*, 2024.

[55] Sudarshan Rajagopalan and Vishal M Patel. Low-rank adaptation-based all-weather removal for autonomous navigation. *arXiv preprint arXiv:2411.17814*, 2024.

[56] J. Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685, 2021.

[57] Manoj Rohit Vemparala, Nael Fasfous, Alexander Frickenstein, Mhd Ali Moraly, Aquib Jamal, Lukas Frickenstein, Christian Unger, Naveen Shankar Nagaraja, and Walter Stechele. L2pf - learning to prune faster. In *International Conference on Computer Vision and Image Processing*, 2021.

[58] Guanghui Qin and Jason Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*, 2021.

[59] SungYeon Park, MinJae Lee, JiHyuk Kang, Hahyeon Choi, Yoonah Park, Juhwan Cho, Adam Lee, and DongKyu Kim. Vlaad: Vision and language assistant for autonomous driving. *2024 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, pages 980–987, 2024.

[60] Yuan Sun, Navid Salami Pargoo, Peter J. Jin, and Jorge Ortiz. Optimizing autonomous driving for safety: A human-centric approach with llm-enhanced rlhf. *ArXiv*, abs/2406.04481, 2024.

[61] Haicheng Liao, Hanlin Kong, Bonan Wang, Chengyue Wang, Wang Ye, Zhengbing He, Chengzhong Xu, and Zhenning Li. Cot-drive: Efficient motion forecasting for autonomous driving with llms and chain-of-thought prompting. 2025.

[62] Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. A language agent for autonomous driving. *arXiv preprint arXiv:2311.10813*, 2023.

[63] Wenhai Wang, Jiangwei Xie, ChuanYang Hu, Haoming Zou, Jianan Fan, Wenwen Tong, Yang Wen, Silei Wu, Hanming Deng, Zhiqi Li, et al. Drivemlm: Aligning multi-modal large language models with behavioral planning states for autonomous driving. *arXiv preprint arXiv:2312.09245*, 2023.

[64] Junzhou Chen and Sidi Lu. An advanced driving agent with the multimodal large language model for autonomous vehicles. In *2024 IEEE International Conference on Mobility, Operations, Services and Technologies (MOST)*, pages 1–11. IEEE, 2024.

[65] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

[66] Yibo Liu, Zheyuan Yang, Guile Wu, Yuan Ren, Kejian Lin, Bingbing Liu, Yang Liu, and Jinjun Shan. Vqa-diff: Exploiting vqa and diffusion for zero-shot image-to-3d vehicle asset generation in autonomous driving. In *European Conference on Computer Vision*, pages 323–340. Springer, 2024.

[67] Goodarz Mehr and Azim Eskandarian. Simbev: A synthetic multi-task multi-sensor driving data generation tool and dataset. *arXiv preprint arXiv:2502.01894*, 2025.

[68] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela L Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In *2023 IEEE international conference on robotics and automation (ICRA)*, pages 2774–2781. IEEE, 2023.

[69] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[70] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[71] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llavanext: Improved reasoning, ocr, and world knowledge, 2024.

[72] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[73] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[74] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[75] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[76] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[77] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[78] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[79] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[80] Hao Sha, Yao Mu, Yuxuan Jiang, Li Chen, Chenfeng Xu, Ping Luo, Shengbo Eben Li, Masayoshi Tomizuka, Wei Zhan, and Mingyu Ding. Languagempc: Large language models as decision makers for autonomous driving. *arXiv preprint arXiv:2310.03026*, 2023.

[81] Shengbo Li, Keqiang Li, Rajesh Rajamani, and Jianqiang Wang. Model predictive multi-objective vehicular adaptive cruise control. *IEEE Transactions on control systems technology*, 19(3):556–566, 2010.

[82] Julong Wei, Shanshuai Yuan, Pengfei Li, Qingda Hu, Zhongxue Gan, and Wenchao Ding. Occllama: An occupancy-language-action generative world model for autonomous driving. *arXiv preprint arXiv:2409.03272*, 2024.

[83] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[84] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[85] Wenzhao Zheng, Weiliang Chen, Yuanhui Huang, Borui Zhang, Yueqi Duan, and Jiwen Lu. Occworld: Learning a 3d occupancy world model for autonomous driving. In *European conference on computer vision*, pages 55–72. Springer, 2024.

[86] Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models. *arXiv preprint arXiv:2402.12289*, 2024.

[87] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. 2023.

[88] Zhijian Huang, Chengjian Feng, Feng Yan, Baihui Xiao, Zequn Jie, Yujie Zhong, Xiaodan Liang, and Lin Ma. Drivemm: All-in-one large multimodal model for autonomous driving. *arXiv preprint arXiv:2412.07689*, 2024.

[89] Yinpeng Liu, Jiawei Liu, Xiang Shi, Qikai Cheng, Yong Huang, and Wei Lu. Let's learn step by step: Enhancing in-context learning ability with curriculum learning. *arXiv preprint arXiv:2402.10738*, 2024.

[90] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[91] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023.

[92] Meta AI. Introducing llama 3.1: Our most capable models to date, 2024.

[93] Jiageng Mao, Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. 3d object detection for autonomous driving: A comprehensive survey. *International Journal of Computer Vision*, 131(8):1909–1963, 2023.

[94] Sergio Casas, Wenjie Luo, and Raquel Urtasun. Intentnet: Learning to predict intention from raw sensor data. In *Conference on Robot Learning*, pages 947–956. PMLR, 2018.

[95] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 523–540. Springer, 2020.

[96] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000.

[97] Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning*, 2023.

[98] Honghui Yang, Tong He, Jiaheng Liu, Hua Chen, Boxi Wu, Binbin Lin, Xiaofei He, and Wanli Ouyang. Gd-mae: generative decoder for mae pre-training on lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9403–9414, 2023.

[99] Ana-Maria Marcu, Long Chen, Jan Hünermann, Alice Karnsund, Benoit Hanotte, Prajwal Chidananda, Saurabh Nair, Vijay Badrinarayanan, Alex Kendall, Jamie Shotton, et al. Lingoqa: Visual question answering for autonomous driving. In *European Conference on Computer Vision*, pages 252–269. Springer, 2024.

[100] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2021.

[101] Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6, 2023.

[102] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18392–18402, 2023.

[103] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3836–3847, 2023.

[104] Dongxu Li, Junnan Li, and Steven Hoi. Blip-diffusion: Pre-trained subject representation for controllable text-to-image generation and editing. *Advances in Neural Information Processing Systems*, 36:30146–30166, 2023.

[105] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[106] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[107] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022.

[108] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[109] Lloyd Russell, Anthony Hu, Lorenzo Bertoni, George Fedoseev, Jamie Shotton, Elahe Arani, and Gianluca Corrado. Gaia-2: A controllable multi-view generative world model for autonomous driving. *arXiv preprint arXiv:2503.20523*, 2025.

[110] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[111] Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kwan-Yee K Wong, Zhenguo Li, and Hengshuang Zhao. Drivegpt4: Interpretable end-to-end autonomous driving via large language model. *IEEE Robotics and Automation Letters*, 2024.

[112] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[113] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1728–1738, 2021.

[114] Jianhao Yuan, Shuyang Sun, Daniel Omeiza, Bo Zhao, Paul Newman, Lars Kunze, and Matthew Gadd. Rag-driver: Generalisable driving explanations with retrieval-augmented in-context learning in multi-modal large language model. *arXiv preprint arXiv:2402.10828*, 2024.

[115] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

[116] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The international journal of robotics research*, 32(11):1231–1237, 2013.

[117] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[118] Zheng Tang, Milind Naphade, Ming-Yu Liu, Xiaodong Yang, Stan Birchfield, Shuo Wang, Ratnesh Kumar, David Anastasiu, and Jenq-Neng Hwang. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8797–8806, 2019.

[119] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[120] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2636–2645, 2020.

[121] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020.

[122] Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. Textual explanations for self-driving vehicles. In *Proceedings of the European conference on computer vision (ECCV)*, pages 563–578, 2018.

[123] Dongming Wu, Wencheng Han, Yingfei Liu, Tiancai Wang, Cheng-zhong Xu, Xiangyu Zhang, and Jianbing Shen. Language prompt for autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 8359–8367, 2025.

[124] Tushar Choudhary, Vikrant Dewangan, Shivam Chandhok, Shubham Priyadarshan, Anushka Jain, Arun K Singh, Siddharth Srivastava, Krishna Murthy Jatavallabhula, and K Madhava Krishna. Talk2bev: Language-enhanced bird's-eye view maps for autonomous driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16345–16352. IEEE, 2024.

[125] Srikanth Malla, Chiho Choi, Isht Dwivedi, Joon Hee Choi, and Jiachen Li. Drama: Joint risk localization and captioning in driving. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1043–1052, 2023.

A Vehicle functions authority at each automation stage

Function	**Level 0 (No automation)**	**Level 1 (Driving assistant)**	**Level 2 (Partial automation)**	**Level 3 (Conditional automation)**	**Level 4 (High automation)**	**Level 5 (Full automation)**
Steering control	Human driver	Human driver	Automation system	Automation system	Automation system	Automation system
Acceleration and Deceleration	Human driver	Automation system	Automation system	Automation system	Automation system	Automation system
Braking	Human driver	Automation system	Automation system	Automation system	Automation system	Automation system
Lane keeping	Human driver	Human driver	Automation system	Automation system	Automation system	Automation system
Lane changing	Human driver	Human driver	Human driver	Automation system	Automation system	Automation system
Navigation and Route planning	Human driver	Human driver	Human driver	Human driver	Automation system	Automation system
Traffic sign and signal recognition	Human driver	Human driver	Automation system	Automation system	Automation system	Automation system
Object detection and response	Human driver	Human driver	Automation system	Automation system	Automation system	Automation system
Parking	Human driver	Human driver	Automation system	Automation system	Automation system	Automation system
Monitoring vehicle system	Human driver	Human driver	Human driver	Automation system	Automation system	Automation system
Emergency response	Human driver	Human driver	Automation system	Automation system	Automation system	Automation system
Adaptive cruise control	Human driver	Automation system	Automation system	Automation system	Automation system	Automation system
Environmental perception	Human driver	Human driver	Human driver	Automation system	Automation system	Automation system
Vehicle-to-everything (V2X) communication	Human driver	Human driver	Human driver	Automation system	Automation system	Automation system
Driver Monitoring (Semi-autonomous Vehicles)	Not applicable	Not applicable	Human driver	Human driver	Not applicable	Not applicable

Table 2: 15 functions of vehicle automation, that indicate who has authority at each stage of automation [1]