DriveLM:
Driving with Graph Visual Question Answering

Chonghao Sima[4,1*] Katrin Renz[2,3*] Kashyap Chitta[2,3] Li Chen[4,1]
Hanxue Zhang[1] Chengen Xie[1] Jens Beißwenger[2,3] Ping Luo[4]
Andreas Geiger[2,3†] Hongyang Li[1†]

[1] OpenDriveLab, Shanghai AI Lab, China
[2] University of Tübingen, Germany
[3] Tübingen AI Center, Germany
[4] University of Hong Kong, China
simachonghao@pjlab.org.cn katrin.renz@uni-tuebingen.de

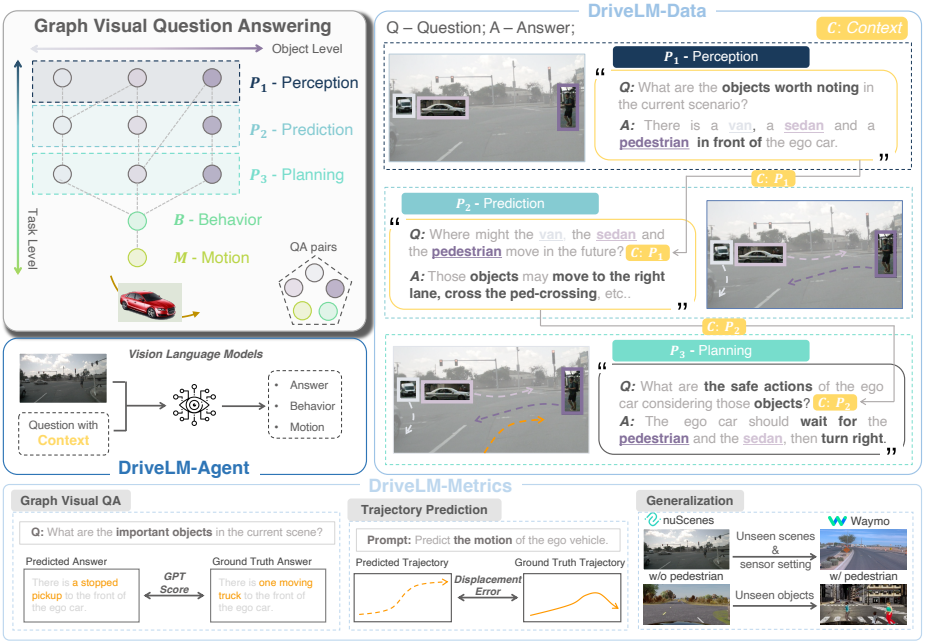

Fig. 1: We present **DriveLM**: A new task, dataset, metrics, and baseline for end-to-end autonomous driving. Inspired by [9], DriveLM considers **Graph Visual Question Answering (GVQA)**, where question-answer pairs are interconnected via logical dependencies at the object-level, *i.e.*, interactions between object pairs, and the task-level, *e.g.*, perception → prediction → planning → behavior (discretized action described in natural language) → motion (continuous trajectory). We propose **DriveLM-Data** for training **DriveLM-Agent**, a baseline for GVQA. We validate its effectiveness using the **DriveLM-Metrics** on challenging settings requiring zero-shot generalization.

Abstract. We study how vision-language models (VLMs) trained on web-scale data can be integrated into end-to-end driving systems to

*Equal contribution. †Project lead.

boost generalization and enable interactivity with human users. While recent approaches adapt VLMs to driving via single-round visual question answering (VQA), human drivers reason about decisions in multiple steps. Starting from the localization of key objects, humans estimate object interactions before taking actions. The key insight is that with our proposed task, Graph VQA, where we model graph-structured reasoning through perception, prediction and planning question-answer pairs, we obtain a suitable proxy task to mimic the human reasoning process. We instantiate datasets (DriveLM-Data) built upon nuScenes and CARLA, and propose a VLM-based baseline approach (DriveLM-Agent) for jointly performing Graph VQA and end-to-end driving. The experiments demonstrate that Graph VQA provides a simple, principled framework for reasoning about a driving scene, and DriveLM-Data provides a challenging benchmark for this task. Our DriveLM-Agent baseline performs end-to-end autonomous driving competitively in comparison to state-of-the-art driving-specific architectures. Notably, its benefits are pronounced when it is evaluated zero-shot on unseen sensor configurations. Our question-wise ablation study shows that the performance gain comes from the rich annotation of prediction and planning QA pairs in the graph structure. All data, models and an official evaluation server are available at https://github.com/OpenDriveLab/DriveLM.

Keywords: Vision Language Model · End-to-end Autonomous Driving

1 Introduction

Current Autonomous Driving (AD) stacks are still lacking crucial capabilities [9, 12]. One key requirement is generalization, which involves the ability to handle unseen scenarios or unfamiliar sensor configurations. A secondary requirement pertains to the interaction of these models with humans, highlighted for example by EU regulations that mandate explainability in deployment [3]. Furthermore, unlike today's AD models, humans do not navigate based on geometrically precise bird's-eye view (BEV) representations [14,25,30]. Instead, humans implicitly perform object-centric perception, prediction, and planning (which we refer to as P_{1-3}): a rough identification and localization of key objects, followed by reasoning about their possible movement and aggregation of this information into a driving action [40, 53].

Simultaneously, another field has been forging ahead: Vision-Language Models (VLMs) [31, 34, 63, 74]. These models have several strengths. First, they hold a base understanding of the world from internet-scale data that could potentially facilitate generalization for planning in AD. In fact, this sort of generalization has already been achieved by VLMs for simpler robotics tasks [18,75]. Second, the use of language representations as an input and output offers a platform for human-friendly interaction with these models, unlike bounding boxes or trajectories that are more common to current methods [15, 24, 32, 48]. Finally, VLMs are able to make decisions in multiple steps linked by logical reasoning [5, 16, 65, 67, 72, 75].

Importantly, even though they reason in multiple separate steps, VLMs are end-to-end differentiable architectures, a characteristic that is highly desirable for autonomous driving [9].

Recent work towards enabling the application of VLMs to AD systems falls into two categories: scene-level or single object-level Visual Question Answering (VQA). Scene-level VQA refers to the task of describing the driving behavior by one or two supporting reasons, *e.g.*, "The car is moving into the right lane because it is safe to do so." [28,29]. Single object-level VQA formulates the understanding of the ego vehicle's response to a single object by a chain of QAs in the form of "what-which-where-how-why", *e.g.*, "The ego vehicle stops because there is a pedestrian in a white shirt crossing the intersection in front of the ego vehicle and it does not want to crash into the pedestrian." [36, 46, 49]. Unfortunately, neither of these paradigms provides a suitable proxy task to mimic the P_{1-3} reasoning process in humans, who consider multiple objects and reason about each in multiple steps. Therefore, in this paper, we propose a new task, along with corresponding datasets and a baseline model architecture (Fig. 1).

Task. Graph Visual Question Answering (GVQA) involves formulating P_{1-3} reasoning as a series of question-answer pairs (QAs) in a directed graph. Its key difference to the aforementioned VQA tasks for AD is the availability of logical dependencies between QAs which can be used to guide the answering process. GVQA also encompasses questions regarding behavior and motion planning, with dedicated metrics (details in Section 2).

Datasets. DriveLM-nuScenes and **DriveLM-CARLA** consist of annotated QAs, arranged in a graph, linking images with driving behavior through logical reasoning. In comparison to existing benchmarks, they provide significantly more text annotations per frame (Table 1 and Fig. 2). Furthermore, as an integral component of DriveLM-CARLA, we build PDM-Lite [4], the first working rule-based expert algorithm for CARLA Leaderboard 2.0. We pair these training datasets with challenging test data for evaluating zero-shot generalization.

Baseline. DriveLM-Agent employs a trajectory tokenizer that can be applied to any general VLM [31,34,44,74], coupled with a graph prompting scheme that models logical dependencies as context inputs for VLMs. The result is a simple methodology to effectively repurpose VLMs for end-to-end AD (Section 3).

Our experiments provide encouraging results. We find that GVQA on DriveLM is a challenging task, where current methods obtain moderate scores and better modeling of logical dependencies is likely necessary to achieve strong QA performance. Even so, DriveLM-Agent already performs competitively to state-of-the-art driving-specific models [25] when tested in the open-loop planning setting, despite its task-agnostic architecture. Furthermore, employing a graph structure improves zero-shot generalization, enabling DriveLM-Agent to correctly handle unseen deployment on Waymo data [54] after training only on nuScenes [6] data. Finally, with a question-wise analysis we find that the QA pairs from the prediction and planning stages help the final driving decision most. From these results, we believe that improving GVQA holds great potential towards building autonomous driving agents with strong generalization.

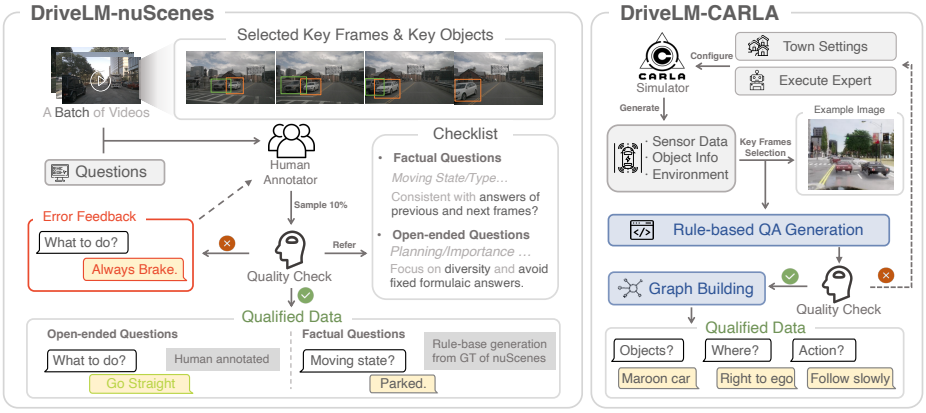

Fig. 2: Annotation Pipeline: In DriveLM-nuScenes, we adopt a semi-rule-based QA labeling pipeline, where both the ground truth annotation in nuScenes/OpenLane-V2 and feedback from human annotators are used. A critical part of our pipeline is the multi-round quality check, which guarantees high data quality at reasonable costs. In DriveLM-CARLA, we meet the same standards while exploiting a fully rule-based QA labeling pipeline instead, using a new expert algorithm called PDM-Lite.

2 DriveLM: Task, Data, Metrics

Human drivers usually decompose their decision-making process into distinct stages that follow a logical progression which encompasses the identification and localization of key objects, their possible future action and interaction, and ego planning based on all this information [20, 35]. This inspires us to propose the GVQA as the critical ingredient of DriveLM, which serves as a suitable proxy task to mimic the human reasoning process. Within this section, we illustrate the formulation of the GVQA task (Section 2.1), introduce DriveLM-Data (Section 2.2) to exemplify the instantiation of GVQA using prominent driving datasets, and overview the DriveLM-Metrics used for evaluation (Section 2.3). To encourage further research in this direction, an official evaluation server (with a public leaderboard) will be set up to benchmark different methods and discover more insights about combining language models with autonomous driving.

2.1 DriveLM-Task: GVQA

We organize all the question-answer pairs (QAs) for an image frame into a graph. We use the terminology "graph-structured" to refer to a directed acyclic graph (DAG), e.g., the current question can get context from (multiple) parent and grandparent nodes. The graph $G = (V, E)$ contains a set of vertices V, where each vertex represents a QA pair $v = (q, a)$ associated with one or more key objects in the scenario. The key difference between GVQA and ordinary VQA is that the QAs in GVQA have logical dependencies, which we formulate as the edges between the vertices. $E \subseteq V \times V$, is a set of directed edges, where each edge $e = (v_p, v_c)$ connects the parent QA and the child QA. We formulate the

Table 1: Comparison of DriveLM-nuScenes & -CARLA with Existing Datasets. DriveLM-Data significant advances annotation quantity, comprehensiveness (covering **perception, prediction and planning**), and logic (chain to **graph**). † full dataset, ‡ keyframe dataset, * semi-rule-based labeling (w/ human annotators), ** fully-rule-based (no human annotators). - means publicly unavailable.

Dataset	Source Dataset	# Frames	Avg. QA per Frame	Perception	Prediction	Planning	Logic
nuScenes-QA [46]	nuScenes	34,149	13.5	460k**	✗	✗	None
nuPrompt [68]	nuScenes	34,149	1.0	35k*	✗	✗	None
HAD [28]	HDD	25,549	1.8	25k	✗	20k	None
BDD-X [29]	BDD	26,228	1	26k	✗	✗	None
LingoQA [39]	LingoQA	28,000	15.3	-	-	-	None
DRAMA [36]	DRAMA	17,785	5.8	85k	✗	17k	Chain
Rank2Tell [49]	Rank2Tell	5,800	-	-	✗	-	Chain
DriveLM-nuScenes	nuScenes	4,871	**91.4**	144k*	153k	146k	**Graph**
DriveLM-CARLA†	CARLA	64,285	24.4	697k**	311k**	558k**	**Graph**
DriveLM-CARLA‡	CARLA	5,721	24.8	63k**	28k**	51k**	**Graph**

edge set E by incorporating two dimensions: object-level and task-level edges. At the object level, we construct the logical edges $e \in E$ to represent the impact of interactions between different objects. For example, the planning QA node for the sedan is influenced by the perception QA node of the pedestrian in the illustration from Fig. 1 (center). At the task-level, we establish the logical edges $e \in E$ to capture the logical chain of different reasoning stages:

- **Perception** (P_1): identification, description, and localization of key objects in the current scene.
- **Prediction** (P_2): estimation of possible action/interaction of key objects based on perception results.
- **Planning** (P_3): possible safe actions of the ego vehicle.
- **Behavior** (B): classification of driving decision.
- **Motion** (M): waypoints of ego vehicle future trajectory.

The concepts of perception, prediction, and planning (P_{1-3}) are similar to those in end-to-end AD [9], while the concepts of motion and behavior are based on the ego vehicle future trajectory. Specifically, we define the motion M as the ego vehicle future trajectory, which is a set of N points with coordinates (x, y) in bird's-eye view (BEV), denoted as $M = \{(x_0, y_0), (x_1, y_1), ..., (x_N, y_N)\}$. Each point is the offset between the future position and the current position by a fixed time interval. Then, the distance for x, y at each time interval is computed as:

$$\{x, y\}_{\text{dist}} = \{(\delta_{x,1}, \delta_{y,1}), ..., (\delta_{x,N}, \delta_{y,N})\}, \tag{1}$$

where $\delta_{x,i} = x_i - x_{i-1}$ and $\delta_{y,i} = y_i - y_{i-1}$, for $i = 1, 2, \ldots, N$. The goal of the behavior representation is to serve as an interface from P_{1-3} to M. To obtain a behavior representation, we map the mean of x_{dist} and y_{dist} to one of the predefined bins, where each bin corresponds to a category in either speed or steering. These are denoted as B_{sp} and B_{st} respectively. In this work, we consider 5 bins:

$$B_{sp} \in \{\texttt{fast}_2, \texttt{fast}_1, \texttt{moderate}, \texttt{slow}_1, \texttt{slow}_2\},$$
$$B_{st} \in \{\texttt{left}_2, \texttt{left}_1, \texttt{straight}, \texttt{right}_1, \texttt{right}_2\},$$

where the number in the subscript indicates the intensity. The combination of the speed and steering categories for a trajectory form its behavior category as $B = (B_{sp}, B_{st})$. While we use a simple definition of B as a starting point for research on driving with VLMs, we note that our formulation supports the incorporation of more abstract behaviors such as a lane changes or overtaking.

2.2 DriveLM-Data

In order to provide comprehensive and accurate QAs with the graph structure defined in Section 2.1, we introduce DriveLM-nuScenes and DriveLM-CARLA. Since there are significant disparities between nuScenes and CARLA, the collection methods and statistics of these datasets differ.

DriveLM-nuScenes. We divide the annotation process into three steps: selecting key frames from video clips, choosing key objects within these key frames, and subsequently annotating the frame-level P_{1-3} QAs for these key objects. A portion of the Perception QAs are generated from the nuScenes [6] and OpenLane-V2 [60] ground truth, while the remaining QAs are manually annotated. The question templates for the manual annotations were designed by 5 domain experts accounting for how humans make driving decisions. Annotators are prompted with all question templates for each frame, encouraged to answer all questions, but provided a skip option to account for possible incompatibility. As we manually annotate the vast majority of data in DriveLM-nuScenes, quality is particularly crucial for this portion. When annotating, we conduct multiple rounds of rigorous quality checks. In each round, we categorize the data into different batches and inspect ten percent of the data in each batch. If the qualification rate of manually annotated data in this ten percent does not meet expectations, we request the annotators to re-label all data in the batch. In Fig. 2, we showcase an example of the QA annotation pipeline, where all questions undergo quality checks according to our standards. As a result, DriveLM-nuScenes stands out from previously proposed datasets with its larger scale, greater comprehensiveness, and more complex structure (See Table 1). These QAs cover various aspects of the driving process, ranging from perception and prediction to planning, providing a comprehensive understanding of autonomous driving scenarios (details in the supplementary material).

DriveLM-CARLA Expert. We collect data using CARLA 0.9.14 in the Leaderboard 2.0 framework [17]. Leaderboard 2.0 contains a large set of new driving scenarios compared to its predecessor, Leaderboard 1.0. However, as of now, there is no existing method to collect training data at scale in Leaderboard 2.0. For example, the privileged rule-based expert used by TransFuser++ [26], a state-of-the-art method in Leaderboard 1.0, obtains a Driving Score (DS) of only 2% on the 8+ kilometer long official validation routes of Leaderboard 2.0. We build a new expert algorithm, PDM-Lite, that can handle the new challenges

in Leaderboard 2.0. Our approach is similar to PDM-Closed [15], a rule-based planner for nuPlan [7]. PDM-Lite uses the Intelligent Driver Model (IDM) [59] to obtain a target speed based on the leading vehicle, pedestrian, stop sign or traffic light. Unlike the 16 proposals from which one is selected via a complex cost function in PDM-Closed, we use only two proposals and a simpler cost function based on the TransFuser++ expert [26], giving a light-weight planner suitable for scalable QA generation. PDM-Lite obtains an improved DS of 44% on the official CARLA validation routes. More details can be found in the supplementary.

DriveLM-CARLA QA Generation. For collecting the DriveLM-CARLA dataset, we set up a series of routes in urban, residential, and rural areas and execute PDM-Lite on these routes. During this process, we collect the necessary sensor data, sample keyframes, generate relevant QAs based on privileged information about objects and the scene, and organize the logical relationships to connect this series of QAs into a graph. We generate data and labels at 4 FPS and extract keyframes based on changes in the decision of the expert (e.g. when the expert changes from acceleration to braking). The rule-based annotation pipeline is illustrated in Fig. 2. During data collection, we extract privileged information from the simulator about the status of the static and dynamic objects in the scene as well as the triggered rules of the expert. We use all 38 scenarios except for InterurbanAdvancedActorFlow, MergerIntoSlowTraffic, and VehicleTurningRoute to create questions and answers with hand-crafted sentence templates based on the information we extract from the simulator. The exact questions with their graph structure can be found in the supplementary. Our annotation process has the advantage of straightforward scalability since we only need to define route and scenario settings in CARLA and the subsequent steps can be executed automatically. Including 1.6M QAs (with a straightforward scaling recipe), our DriveLM-CARLA stands out as the largest driving-language benchmark in terms of total textual content among existing benchmarks as shown in Table 1.

2.3 DriveLM-Metrics

To evaluate GVQA, the DriveLM-Metrics consist of three components for evaluating motion M, behavior B, and P_{1-3}. For measuring the performance of the motion stage, we use standard metrics from the nuScenes and Waymo benchmarks: average and final displacement error, (**ADE, FDE**), and the **collision rate** on the predicted trajectory, following UniAD [25]. We evaluate behavior predictions by the **classification accuracy**, along with a breakdown of the overall accuracy into its steering and speed components. Finally, we measure the P_{1-3}performance using two metrics. **SPICE** [2] is a prevailing metric used in VQA and image captioning, which calculates the structure similarity of predicted texts with ground truth while ignoring the semantic meanings. Simultaneously, we employ **GPT Score** to measure the semantic alignment of answers and complement the SPICE metric. Specifically, the question, the ground truth answer, the predicted answer, and a prompt asking for a numerical score of the answer

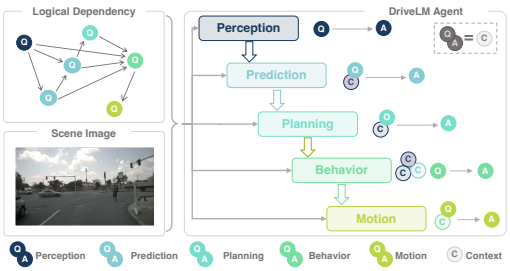

Fig. 3: DriveLM-Agent Pipeline. Given the scene image, a VLM performs prompting with context to model the logical dependency among the five QA stages. Context is built using preceding QAs, and can have one or more sources.

are sent to ChatGPT-3.5 [41, 42]. We parse the text returned to get the score, where a higher score indicates better semantic accuracy.

3 DriveLM-Agent: A GVQA Baseline

In this section, we present DriveLM-Agent, a baseline approach for the GVQA task detailed in Section 2. DriveLM-Agent is built upon a general vision-language model and can therefore exploit underlying knowledge gained during pre-training. Our overall goal involves translating an image into the desired ego vehicle motion (M) through the different stages of VQA (P_1, P_2, P_3, B). For this, we choose BLIP-2 [31] as our base VLM due to its simplicity in architecture and flexibility in fine-tuning, but our approach can be applied agnostically to other VLMs.

As shown in Fig. 3, DriveLM-Agent can be decomposed into several stages: (1) P_{1-3}, *i.e.*, *perception, prediction, planning*, serve as the foundational layers to understand the scene and reason about its structure. (2) The *behavior* stage aggregates crucial information from the P_{1-3} into a description of the desired driving action in language space. (3) Finally, the *motion* stage translates the behavior into an executable driving trajectory. To implement the logical dependency between each linked QA, we propose to use context between connected nodes in the GVQA graph. We expand on this idea in the following.

3.1 Prompting with Context

Directly translating images to motion as in [13, 45] is extremely challenging. Motivated by the tendency of humans to perform a multi-step reasoning process, we propose to use a similar strategy for VLM-based driving. By doing so, we facilitate the retrieval of knowledge stored in LLMs and improve explainability.

More precisely, the model is designed to use answers from the previous steps in the reasoning process as the context for the following questions. For each edge $e = (v_p, v_c) \in E$, we append the QA from the parent node v_p to the question of the current node v_c with a prefix "*Context:* ". The context can also contain QAs from multiple preceding nodes in which case we concatenate all QAs to one context sequence. It is worth noting that the context is only one possible implementation to formulate logical dependency in GVQA, which we select due to its simplicity. With this scheme, we pass forward relevant information based on the logical dependencies established by the graph.

Note that the size and structure of the graph during inference is a design choice of the algorithm, which can be adapted based on the task or available compute budget. We use this property to train on all available QAs, but perform inference on specific subgraphs, where the questions are sampled using heuristics. For more details, please refer to the supplementary material.

Context Aggregation through Behavior. Driving encompasses a wide array of potential situations that require an appropriate response. However, despite the diversity of these circumstances, it is interesting to note that almost all events involve decisions that can be discretized into a set of behaviors. For example, applying the brakes appropriately may address various situations such as a red light signal, a stop sign, or the presence of an object ahead of the vehicle. The focus of our behavior stage is to generate such a behavior: a statement in natural language that articulates the vehicle's intended movement. As described in Section 2.1, this description effectively serves as a reflective step wherein the model summarizes all crucial information from the graph. Thus, we propose to use all possible sources of context for predicting behavior, *i.e.*, all the QAs in P_{1-3}. We empirically observe that this design is crucial for driving with VLMs.

3.2 Trajectory Tokenization for Motion

Since it is non-trivial to output fine-grained numerical results using general VLMs, RT-2 [75] handles robotic actions based on a specialized trajectory tokenization module. We use this approach to enable DriveLM-Agent to take as input the image and behavior description and output trajectories. Specifically, we divide the coordinates of waypoints into 256 bins empirically based on statistics of train set trajectories. We re-define tokens in the BLIP-2 language tokenizer, establishing tokens for each bin, and fine-tune the VLM on the redefined vocabulary. For simplicity, we use the same VLM architecture (BLIP-2) to perform this task, but with independent LoRA weights and trained on a dataset consisting of only the QAs for this motion stage. Thus, it is possible to perform this functionality using a lightweight LLM [47] or driving-specific architecture that accepts a command as an input [24, 69].

4 Experiments

In this section, we present our experimental results that aim to address the following research questions: (1) How can VLMs be effectively repurposed for end-to-end autonomous driving? (2) Can VLMs for driving generalize when evaluated with unseen sensor setups? (3) What is the effect of each question in perception, prediction and planning on the final behavior decision? (4) How well do VLMs perform perception, prediction, and planning via GVQA? Experiments about more VLMs on DriveLM-nuScenes and generalization to unseen objects on DriveLM-CARLA are included in the supplementary material.

Setup. We now briefly overview the key implementation details for the two settings used in our experiments (additional details are provided in the supple-

mentary material). All fine-tuning is implemented with LoRA [23]. On DriveLM-nuScenes, we finetune BLIP-2 on the `train` split for 10 epochs. We use a batch size of 2 for each GPU, and the entire training process spans approximately 7 hours with 8 V100 GPUs. We train BLIP-2 on the keyframes of the `train` split of DriveLM-CARLA for 6 epochs. This takes 6 hours on 4 A100 GPUs.

4.1 VLMs for End-to-End Driving

In our first experiment, we aim to assess the ability of VLMs to perform open-loop planning on DriveLM-nuScenes. In particular, we investigate the impact of the context provided to the behavior and motion stages. Given sensor data (and in the case of VLM methods, a text input), the model is required to predict the ego-vehicle future trajectory in the form of waypoints. Though open-loop planning suffers from a mismatched data distribution [33], we try to alleviate this by evaluating on only the key frames annotated in DriveLM-nuScenes.

Baselines. As a reference for the task's difficulty, we provide a simple **Command Mean** baseline. Each frame in nuScenes is associated with one of 3 commands, 'turn left', 'turn right', or 'go straight'. We output the mean of all trajectories in the training set whose command matches the current test frame command. Further, we compare our approach to the current state-of-the-art on nuScenes, UniAD [25]. Besides its original setting which requires video inputs, we train a single-frame version ('**UniAD-Single**') for a fair comparison. Finally, **BLIP-RT-2** denotes BLIP-2 [31] fine-tuned on DriveLM-Data with the trajectory tokenization scheme described in Section 3.2 for only the motion task. This acts as an indicator for the performance when using an identical network architecture as DriveLM-Agent, but no context inputs or VQA training data.

DriveLM-Agent. We consider 3 variants of DriveLM-Agent incorporating our proposed changes in steps: (1) a 2-stage version that predicts behavior and then motion (as described in Section 2.1), but without any P_{1-3} context for behavior prediction ('None'); (2) a 'Chain' version that builds the P_{1-3} graph, but only passes the final node (P_3) to the behavior stage; (3) the full model ('Graph') that uses all QAs from P_{1-3} as context for B.

Results. We show results for the above methods in Table 2. Among the baselines, BLIP-RT-2 is unable to match UniAD-Single (though both perform well relative to Command Mean). This shows that the single-stage approach without any reasoning is unable to compete with the prior state-of-the-art on nuScenes. However, the proposed DriveLM-Agent, which predicts behavior as an intermediate step for motion, provides a significant boost in performance, surpassing UniAD-Single. This indicates that with the appropriate prompting, VLMs can be surprisingly competitive for end-to-end driving. Interestingly, in the experimental settings that do not involve generalization, the Chain and Graph versions of DriveLM-Agent do not provide any further advantage over no context. Further, single-frame VLMs fall short compared to the privileged video-based UniAD model, indicating that VLMs with video inputs may be necessary for this task. We provide video-input VLM result in the supplementary material.

Table 2: Open-loop Planning on DriveLM-nuScenes and Zero-shot Generalization across Sensor Configurations on Waymo. Under nuScenes, using Behavior (B) as context for Motion (M) enables end-to-end driving with VLMs on par with UniAD-Single, a state-of-the-art driving-specific architecture. Under Waymo, we randomly sampled 1k frames from the Waymo `val` set after training on DriveLM-nuScenes. DriveLM-Agent outperforms UniAD-Single and benefits from graph context.

Method	Behavior (B) Context	Motion (M) Context	nuScenes Behavior (B) Acc. ↑	Speed ↑	Steer ↑	nuScenes Motion (M) ADE ↓	Col. ↓	Waymo Behavior (B) Acc. ↑	Speed ↑	Steer ↑	Waymo Motion (M) ADE ↓	Col. ↓
Command Mean	-	-	-	-	-	4.57	5.72	-	-	-	7.98	11.41
UniAD-Single	-	-	-	-	-	1.80	2.62	-	-	-	4.16	9.31
BLIP-RT-2	-	-	-	-	-	2.63	2.77	-	-	-	2.78	6.47
DriveLM-Agent	None	B	**61.45**	**72.20**	**84.73**	**1.39**	**1.67**	35.70	43.90	65.20	2.76	6.59
	Chain	B	50.43	60.32	75.34	2.07	2.08	34.62	41.28	64.55	2.85	6.89
	Graph	B	57.49	69.89	80.63	1.74	1.89	**39.73**	**54.29**	**70.35**	**2.63**	**6.17**
UniAD [25]	-	-	-	-	-	*0.80*	*0.17*	-	-	-	-	-

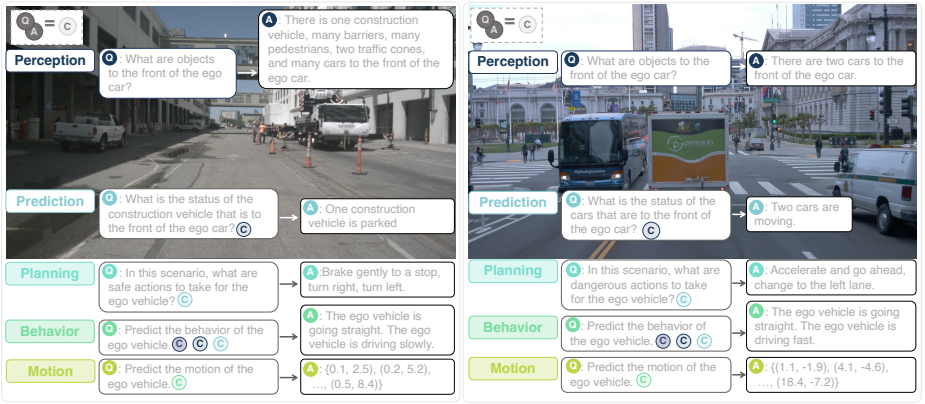

Fig. 4: Qualitative Results of DriveLM-Agent. (Left) DriveLM-nuScenes `val` frame, (Right) Waymo `val` frame. We show the questions (Q), context (C), and predicted answers (A). DriveLM-Agent's outputs are easy to interpret for human users.

4.2 Generalization Across Sensor Configurations

As a more challenging setting for evaluating the models from Section 4.1, we now apply them without further training to a new domain: the Waymo dataset [54]. Waymo does not include rear cameras' images, so we drop this input for UniAD-Single. VLM methods only use the front view and do not require any adaptation.

Results. As shown in Table 2, UniAD-Single does not cope well with the new sensor configuration, and drops below BLIP-RT-2 in performance. The multi-stage approach of DriveLM-Agent provides further improvements. In particular, the accuracy of speed predictions rises from 43.90 with no context to 54.29 with the full graph. On the other hand, the chain approach does not provide sufficient useful information, with a speed accuracy of only 41.28.

We present qualitative results for DriveLM-Agent on nuScenes and Waymo in Fig. 4. The model generally provides intuitive answers, with a few exceptions (*e.g.*, planning on DriveLM-nuScenes, perception on Waymo). This shows the

utility of GVQA towards interactive driving systems. Further, on Waymo, we see meaningful prediction and planning answers despite the imperfect perception. For more visualizations, please see the supplementary material.

4.3 Question-wise Analysis in DriveLM-nuScenes

Next, we analyze the effect of each type of QA pair on the behavior performance by adding them as context. First, a set of representative QA pairs are selected in each stage. We then train BLIP-2 on different combinations of sets of QA pairs, and those QA pairs are added as the context for the behavior question (Table 3).

Representative Questions at each stage. We select different sets of representative QA pairs in each stage as follows:

- P_{1-1}: What are the important objects in the current scene?
- P_{1-2}: What is the moving status of object X?
- P_{1-3}: What is the visual description of object X?
- P_{2-1}: What is the future state of object X?
- P_{2-2}: Would object X be in the moving direction of the ego vehicle?
- P_{2-3}: What object should the ego vehicle notice first / second / third when the ego vehicle is getting to the next possible location?
- P_{3-1}: What actions could the ego vehicle take based on the observation of object X?
- P_{3-2}: What actions taken by the ego vehicle can lead to a collision with object X?
- P_{3-3}: In this scenario, what are safe actions to take for the ego vehicle?

The reasons for such a selection are **1)** statistically they make up about 60% of the total QA pairs in the three stages. **2)** subjectively they represent the most information needed for human to drive. A coarser stage-wise analysis is provided in the supplementary material for further investigation.

Results. Our results are shown in Table 3. We observe that training with QA pairs from prediction and planning stage (**ID** 4-9) improves the performance from training with perception QA pairs only (**ID** 1-3). Adding QA pairs from the planning stage (**ID** 7-9) does not significantly boost the performance compared to their previous stages (**ID** 4-6). The reason could be that other vehicles' future status contains all the necessary information to make the behavior decision.

4.4 Performance for P_{1-3} via GVQA

In our final experiment, we establish baseline results for the P_{1-3} stages of GVQA, studying the impact of context. We use two VLMs, the off-the-shelf BLIP-2 [31] (not fine-tuned on DriveLM), and the proposed DriveLM-Agent.

Baselines. We consider the lower bound of no context ('None'), which corresponds to training and evaluation with the same setting as standard VQA (image and question in, answer out). As an upper bound for each architecture, we perform GVQA but input the ground truth ('GT') context to the model at test time instead of its own prior predictions.

Results. Our results are summarized in Table 4. It can be observed that DriveLM-nuScenes is more challenging for both models, as indicated by the lower scores on

Table 3: Question-wise analysis in DriveLM-nuScenes. Questions of P_{x-y} listed in Section 4.3. Using Prediction and Planning stages of QA pairs as context for Behavior question improves the performance from using Perception only. However, the performance of different questions in the identical stage differentiate slightly.

ID	Perception			Prediction			Planning			Behavior		
	P_{1-1}	P_{1-2}	P_{1-3}	P_{2-1}	P_{2-2}	P_{2-3}	P_{3-1}	P_{3-2}	P_{3-3}	Acc. ↑	Speed ↑	Steer ↑
1	✓	-	-	-	-	-	-	-	-	54.69	66.83	75.22
2	✓	✓	-	-	-	-	-	-	-	55.32	67.33	74.34
3	✓	✓	✓	-	-	-	-	-	-	53.94	65.33	75.72
4	✓	✓	✓	✓	-	-	-	-	-	58.82	71.83	**80.98**
5	✓	✓	✓	✓	✓	-	-	-	-	57.07	71.96	78.97
6	✓	✓	✓	✓	✓	✓	-	-	-	57.70	72.22	79.22
7	✓	✓	✓	✓	✓	✓	✓	-	-	**58.95**	72.59	80.23
8	✓	✓	✓	✓	✓	✓	✓	✓	-	57.95	**72.97**	79.97
9	✓	✓	✓	✓	✓	✓	✓	✓	✓	57.49	69.89	80.63

Table 4: Baseline P_{1-3} Results. DriveLM-Agent and the zero-shot BLIP-2 benefit from a step-wise reasoning procedure given by our graph structure.

Context	**DriveLM-nuScenes** (P_{1-3})				**DriveLM-CARLA** (P_{1-3})			
	BLIP-2 [31]		DriveLM-Agent		BLIP-2 [31]		DriveLM-Agent	
	SPICE ↑	GPT ↑	SPICE ↑	GPT ↑	SPICE ↑	GPT ↑	SPICE ↑	GPT ↑
None	4.34	42.97	42.56	71.39	10.46	46.37	72.71	79.67
Graph	7.71	45.21	49.54	72.51	10.30	55.03	75.26	81.78
GT	*8.19*	*41.10*	*50.29*	*72.94*	*16.18*	*57.98*	*79.07*	*83.13*

it relative to DriveLM-CARLA in all context settings. This is likely due to the higher diversity in human answers obtained for DriveLM-nuScenes, as opposed to the rule-based generation in CARLA. On both datasets, DriveLM-Agent, which is fine-tuned on DriveLM, significantly outperforms BLIP-2 which is applied in a zero-shot manner. On both datasets and cases (zero-shot and fine-tuned), we see the potential of the graph-based structure. For the exact evaluation setting and a question-wise performance analysis, we refer to the supplementary.

5 Related Work

Generalization in Autonomous Driving. The inadequacy of generalization to the "long tail" of corner cases poses significant safety concerns to AD systems [9, 56, 57]. To tackle this issue, prior research primarily makes efforts in data-driven methods [1, 8, 22, 55, 61]. For example, TrafficSim [55] collects more data for safety-critical cases by simulation. An emerging direction involves leveraging semantic information to supervise the detection of unseen or anomalous objects [11, 19, 43, 62]. Even so, the zero-shot performance of AD systems is currently not satisfactory. In this paper, we bring a new approach towards better generalization: learning logical reasoning using Graph VQA.

Language-grounded Driving. Several concurrent methods attempt to incorporate multi-modal inputs into LLMs for AD tasks [10, 11, 21, 27, 37–39, 43, 50, 51,

58, 62, 64, 66, 71, 73]. Specifically, GPT-Driver [37] and LLM-Driver [10] encode the perceived scene state into prompts, relying on LLMs to formulate reasonable plans. DriveGPT4 [71] projects raw sensor data into tokens and utilizes LLMs for end-to-end prediction of control signals and explanations. Despite these preliminary attempts, there is untapped potential in addressing generalization in AD through LLMs. Our work combines VLMs with training over graph-structured QAs from DriveLM. This enables us to show benefits on zero-shot end-to-end planning, which was not demonstrated by these concurrent studies.

6 Discussion

Even though DriveLM exhibits promising generalization, there are concerning limitations of this work.

Table 5: Compared to UniAD-Single, DriveLM-Agent has fewer trainable parameters but a higher inference cost.

Method	#Params	#Trainable	FLOPs	FPS
UniAD-Single	131.9M	58.8M	1.7T	1.8
DriveLM-Agent	3.955B	12.9M	24.2T	0.16

- **Efficiency Constraints.** Inheriting the drawbacks of LLMs, our baseline model suffers from long inference times, especially as we require multiple rounds of predictions based on the graph structure (roughly 10× slower than UniAD, as shown in Table 5). The core problem lies in the slow throughput of the current model (**8.5 tokens/s** in DriveLM-Agent), which may impact practical usage. However, LLMs have become orders of magnitude faster [52, 70] in the last months as this is a general topic of broad interest. We believe that the rapid progress in orthogonal research can alleviate this issue.
- **Driving-specific Inputs.** DriveLM-Agent directly applies the VLM's vision module, taking a low-resolution front-view image as input. Driving-specific sensors such as LiDAR cannot be processed as well. This results in a lack of temporal information and 360-degree scene understanding. Extending DriveLM-Agent to images from multiple views is straightforward as the graph formulation allows various input frames for different nodes. We leave it for future work to explore options for multi-modal and multi-frame inputs.
- **Closed-loop Planning.** Our approach is currently evaluated under an open-loop scheme. In this setting, incorporating the ego vehicle's status as input can significantly enhance the metrics, but its effectiveness may not translate well to the real world, and hence we only consider methods that do not do so. Extending our work to a closed-loop setting with an affordable budget in training time and computational cost is a promising direction to explore. With the usage of CARLA we provide a promising foundation for more research in the direction of closed-loop planning with VLMs.

Conclusion. We show how VLMs can be leveraged as end-to-end autonomous driving agents with improved generalization over task-specific driving stacks. For this we propose the task of Graph VQA together with new datasets and metrics. Equipped with these tools, we build a baseline approach that has a simple architecture and obtains promising results.

Acknowledgements

The OpenDriveLab team is part of the Shanghai AI Lab and kindly supported by National Key R&D Program of China (2022ZD0160104) and NSFC (62206172). This paper is partially supported by the National Key R&D Program of China No.2022ZD0161000 and the General Research Fund of Hong Kong No.17200622 and 17209324. This work was also supported by the BMBF (Tübingen AI Center, FKZ: 01IS18039A), the DFG (SFB 1233, TP 17, project number: 276693517), and by EXC (number 2064/1 – project number 390727645). We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting K. Renz and K. Chitta. Our gratitude goes to Tai Wang for the valuable feedback, Qingwen Bu for refining the figures, and Jiazhi Yang, Shenyuan Gao, Yihang Qiu, Tianyu Li, Yunsong Zhou, Zetong Yang, Julian Zimmerlin for the fruitful discussion.

References

1. Akhauri, S., Zheng, L.Y., Lin, M.C.: Enhanced transfer learning for autonomous driving with systematic accident simulation. In: IROS (2020)
2. Anderson, P., Fernando, B., Johnson, M., Gould, S.: Spice: Semantic propositional image caption evaluation. In: ECCV (2016)
3. Atakishiyev, S., Salameh, M., Babiker, H., Goebel, R.: Explaining autonomous driving actions with visual question answering. arXiv preprint arXiv:2307.10408 (2023)
4. Beißwenger, J.: PDM-Lite: A rule-based planner for carla leaderboard 2.0. https://github.com/OpenDriveLab/DriveLM/blob/DriveLM-CARLA/docs/report.pdf (2024)
5. Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Gianinazzi, L., Gajda, J., Lehmann, T., Podstawski, M., Niewiadomski, H., et al.: Graph of Thoughts: Solving Elaborate Problems with Large Language Models. arXiv preprint arXiv:2308.09687 (2023)
6. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuScenes: A multimodal dataset for autonomous driving. In: CVPR (2020)
7. Caesar, H., Kabzan, J., Tan, K.S., Fong, W.K., Wolff, E.M., Lang, A.H., Fletcher, L., Beijbom, O., Omari, S.: nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. In: CVPR Workshops (2021)
8. Chen, D., Krähenbühl, P.: Learning from all vehicles. In: CVPR (2022)
9. Chen, L., Wu, P., Chitta, K., Jaeger, B., Geiger, A., Li, H.: End-to-end autonomous driving: Challenges and frontiers. arXiv preprint arXiv:2306.16927 (2023)
10. Chen, L., Sinavski, O., Hünermann, J., Karnsund, A., Willmott, A.J., Birch, D., Maund, D., Shotton, J.: Driving with llms: Fusing object-level vector modality for explainable autonomous driving. arXiv preprint arXiv:2310.01957 (2023)
11. Chen, Y., Tonkens, S., Pavone, M.: Categorical traffic transformer: Interpretable and diverse behavior prediction with tokenized latent. arXiv preprint arXiv:2311.18307 (2023)
12. Chib, P.S., Singh, P.: Recent advancements in end-to-end autonomous driving using deep learning: A survey. IEEE T-IV (2023)

13. Chitta, K., Prakash, A., Geiger, A.: Neat: Neural attention fields for end-to-end autonomous driving. In: ICCV (2021)
14. Chitta, K., Prakash, A., Jaeger, B., Yu, Z., Renz, K., , Geiger, A.: Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. IEEE T-PAMI (2023)
15. Dauner, D., Hallgarten, M., Geiger, A., Chitta, K.: Parting with misconceptions about learning-based vehicle motion planning. In: CoRL (2023)
16. Ding, R., Zhang, C., Wang, L., Xu, Y., Ma, M., Zhang, W., Qin, S., Rajmohan, S., Lin, Q., Zhang, D.: Everything of thoughts: Defying the law of penrose triangle for thought generation. arXiv preprint arXiv:2311.04254 (2023)
17. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An open urban driving simulator. In: CoRL (2017)
18. Driess, D., Xia, F., Sajjadi, M.S.M., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q., et al.: PaLM-E: An embodied multimodal language model. In: ICML (2023)
19. Elhafsi, A., Sinha, R., Agia, C., Schmerling, E., Nesnas, I.A., Pavone, M.: Semantic anomaly detection with large language models. Auton. Robot (2023)
20. Groeger, J.A.: Understanding driving: Applying cognitive psychology to a complex everyday task. Routledge (2013)
21. Han, W., Guo, D., Xu, C.Z., Shen, J.: DME-Driver: Integrating human decision logic and 3d scene perception in autonomous driving. arXiv preprint arXiv:2401.03641 (2024)
22. Hanselmann, N., Renz, K., Chitta, K., Bhattacharyya, A., Geiger, A.: King: Generating safety-critical driving scenarios for robust imitation via kinematics gradients. In: ECCV (2022)
23. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-rank adaptation of large language models. In: CoRL (2021)
24. Hu, S., Chen, L., Wu, P., Li, H., Yan, J., Tao, D.: St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In: ECCV (2022)
25. Hu, Y., Yang, J., Chen, L., Li, K., Sima, C., Zhu, X., Chai, S., Du, S., Lin, T., et al.: Planning-oriented autonomous driving. In: CVPR (2023)
26. Jaeger, B., Chitta, K., Geiger, A.: Hidden biases of end-to-end driving models. In: ICCV (2023)
27. Jin, B., Liu, X., Zheng, Y., Li, P., Zhao, H., Zhang, T., Zheng, Y., Zhou, G., Liu, J.: Adapt: Action-aware driving caption transformer. In: ICRA (2023)
28. Kim, J., Misu, T., Chen, Y.T., Tawari, A., Canny, J.: Grounding human-to-vehicle advice for self-driving vehicles. In: CVPR (2019)
29. Kim, J., Rohrbach, A., Darrell, T., Canny, J., Akata, Z.: Textual explanations for self-driving vehicles. In: ECCV (2018)
30. Li, H., Sima, C., Dai, J., Wang, W., Lu, L., Wang, H., Zeng, J., Li, Z., Yang, J., Deng, H., et al.: Delving into the devils of bird's-eye-view perception: A review, evaluation and recipe. IEEE T-PAMI (2023)
31. Li, J., Li, D., Savarese, S., Hoi, S.: BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In: ICML (2023)
32. Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., Dai, J.: Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In: ECCV (2022)
33. Li, Z., Yu, Z., Lan, S., Li, J., Kautz, J., Lu, T., Alvarez, J.M.: Is ego status all you need for open-loop end-to-end autonomous driving? (2023)
34. Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning. In: NeurIPS (2023)

35. Macadam, C.C.: Understanding and modeling the human driver. Veh. Syst. Dyn (2003)
36. Malla, S., Choi, C., Dwivedi, I., Choi, J.H., Li, J.: DRAMA: Joint risk localization and captioning in driving. In: WACV (2023)
37. Mao, J., Qian, Y., Zhao, H., Wang, Y.: GPT-Driver: Learning to drive with gpt. arXiv preprint arXiv:2310.01415 (2023)
38. Mao, J., Ye, J., Qian, Y., Pavone, M., Wang, Y.: A language agent for autonomous driving. arXiv preprint arXiv:2311.10813 (2023)
39. Marcu, A.M., Chen, L., Hünermann, J., Karnsund, A., Hanotte, B., Chidananda, P., Nair, S., Badrinarayanan, V., Kendall, A., et al.: LingoQA: Video question answering for autonomous driving. arXiv preprint arXiv:2312.14115 (2023)
40. Marr, D.: Vision: A computational investigation into the human representation and processing of visual information. The MIT Press (2010)
41. OpenAI: OpenAI: Introducing ChatGPT. https://openai.com/blog/chatgpt (2022)
42. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., et al.: Training language models to follow instructions with human feedback. In: NeurIPS (2022)
43. Pan, C., Yaman, B., Nesti, T., Mallik, A., Allievi, A.G., Velipasalar, S., Ren, L.: VLP: Vision language planning for autonomous driving. arXiv preprint arXiv:2401.05577 (2024)
44. Peng, Z., Wang, W., Dong, L., Hao, Y., Huang, S., Ma, S., Wei, F.: Kosmos-2: Grounding multimodal large language models to the world. arXiv preprint arXiv:2306.14824 (2023)
45. Prakash, A., Chitta, K., Geiger, A.: Multi-modal fusion transformer for end-to-end autonomous driving. In: CVPR (2021)
46. Qian, T., Chen, J., Zhuo, L., Jiao, Y., Jiang, Y.G.: NuScenes-QA: A multi-modal visual question answering benchmark for autonomous driving scenario. arXiv preprint arXiv:2305.14836 (2023)
47. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI blog (2019)
48. Renz, K., Chitta, K., Mercea, O.B., Koepke, A.S., Akata, Z., Geiger, A.: Plant: Explainable planning transformers via object-level representations. In: CoRL (2022)
49. Sachdeva, E., Agarwal, N., Chundi, S., Roelofs, S., Li, J., Dariush, B., Choi, C., Kochenderfer, M.: Rank2Tell: A multimodal driving dataset for joint importance ranking and reasoning. arXiv preprint arXiv:2309.06597 (2023)
50. Seff, A., Cera, B., Chen, D., Ng, M., Zhou, A., Nayakanti, N., Refaat, K.S., Al-Rfou, R., Sapp, B.: MotionLM: Multi-agent motion forecasting as language modeling. In: ICCV (2023)
51. Shao, H., Hu, Y., Wang, L., Waslander, S.L., Liu, Y., Li, H.: LMDrive: Closed-loop end-to-end driving with large language models. arXiv preprint arXiv:2312.07488 (2023)
52. Shi, D., Tao, C., Rao, A., Yang, Z., Yuan, C., Wang, J.: Crossget: Cross-guided ensemble of tokens for accelerating vision-language transformers (2023)
53. Spelke, E.S., Kinzler, K.D.: Core knowledge. Dev Sci (2007)
54. Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al.: Scalability in perception for autonomous driving: Waymo open dataset. In: CVPR (2020)
55. Suo, S., Regalado, S., Casas, S., Urtasun, R.: Trafficsim: Learning to simulate realistic multi-agent behaviors. In: CVPR (2021)
56. Tampuu, A., Matiisen, T., Semikin, M., Fishman, D., Muhammad, N.: A survey of end-to-end driving: Architectures and training methods. IEEE T-NNLS (2020)

57. Teng, S., Hu, X., Deng, P., Li, B., Li, Y., Ai, Y., Yang, D., Li, L., Xuanyuan, Z., Zhu, F., et al.: Motion planning for autonomous driving: The state of the art and future perspectives. IEEE T-IV (2023)
58. Tian, X., Gu, J., Li, B., Liu, Y., Hu, C., Wang, Y., Zhan, K., Jia, P., Lang, X., Zhao, H.: DriveVLM: The convergence of autonomous driving and large vision-language models. arXiv preprint arXiv:2402.12289 (2024)
59. Treiber, M., Hennecke, A., Helbing, D.: Congested traffic states in empirical observations and microscopic simulations. Physical Review E **62**(2) (2000)
60. Wang, H., Li, T., Li, Y., Chen, L., Sima, C., Liu, Z., Wang, B., Jia, P., Wang, Y., Jiang, S., et al.: OpenLane-V2: A topology reasoning benchmark for unified 3d HD mapping. In: NeurIPS Datasets and Benchmarks (2023)
61. Wang, J., Pun, A., Tu, J., Manivasagam, S., Sadat, A., Casas, S., Ren, M., Urtasun, R.: Advsim: Generating safety-critical scenarios for self-driving vehicles. In: CVPR (2021)
62. Wang, T.H., Maalouf, A., Xiao, W., Ban, Y., Amini, A., Rosman, G., Karaman, S., Rus, D.: Drive anywhere: Generalizable end-to-end autonomous driving with multi-modal foundation models. arXiv preprint arXiv:2310.17642 (2023)
63. Wang, W., Chen, Z., Chen, X., Wu, J., Zhu, X., Zeng, G., Luo, P., Lu, T., Zhou, J., Qiao, Y., Dai, J.: VisionLLM: Large language model is also an open-ended decoder for vision-centric tasks. arXiv preprint arXiv:2305.11175 (2023)
64. Wang, W., Xie, J., Hu, C., Zou, H., Fan, J., Tong, W., Wen, Y., Wu, S., Deng, H., et al.: DriveMLM: Aligning multi-modal large language models with behavioral planning states for autonomous driving. arXiv preprint arXiv:2312.09245 (2023)
65. Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., Zhou, D.: Self-Consistency improves chain of thought reasoning in language models. In: ICLR (2023)
66. Wayve: Lingo-1. https://wayve.ai/thinking/lingo-natural-language-autonomous-driving/ (2023)
67. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., Zhou, D.: Chain-of-thought prompting elicits reasoning in large language models. In: NeurIPS (2022)
68. Wu, D., Han, W., Wang, T., Liu, Y., Zhang, X., Shen, J.: Language prompt for autonomous driving. arXiv preprint arXiv:2309.04379 (2023)
69. Wu, P., Jia, X., Chen, L., Yan, J., Li, H., Qiao, Y.: Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. In: NeurIPS (2022)
70. Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., Han, S.: SmoothQuant: Accurate and efficient post-training quantization for large language models. In: Proceedings of the 40th International Conference on Machine Learning (2023)
71. Xu, Z., Zhang, Y., Xie, E., Zhao, Z., Guo, Y., Wong, K.K., Li, Z., Zhao, H.: DriveGPT4: Interpretable end-to-end autonomous driving via large language model. arXiv preprint arXiv:2310.01412 (2023)
72. Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T.L., Cao, Y., Narasimhan, K.: Tree of Thoughts: Deliberate problem solving with large language models. arXiv preprint arXiv:2305.10601 (2023)
73. Yuan, J., Sun, S., Omeiza, D., Zhao, B., Newman, P., Kunze, L., Gadd, M.: RAG-Driver: Generalisable driving explanations with retrieval-augmented in-context learning in multi-modal large language model. arXiv preprint arXiv:2402.10828 (2024)

74. Zhang, R., Han, J., Zhou, A., Hu, X., Yan, S., Lu, P., Li, H., Gao, P., Qiao, Y.: LLaMA-Adapter: Efficient fine-tuning of language models with zero-init attention. arXiv preprint arXiv:2303.16199 (2023)
75. Zitkovich, B., Yu, T., Xu, S., Xu, P., Xiao, T., Xia, F., Wu, J., Wohlhart, P., Welker, S., et al.: RT-2: Vision-language-action models transfer web knowledge to robotic control. In: CoRL (2023)