

DRIVE VLM: The Convergence of Autonomous Driving and Large Vision-Language Models

Supplementary Material

Anonymous Author(s)

Affiliation

Address

email

1 A SUP-AD Dataset

2 A.1 Meta-actions

3 **Meta-action statistics.** We use the meta-action sequence to formally represent the driving strategy.
4 Meta actions are classified into 17 categories. We show the distribution of each meta-action being
5 the first/second/third place in the meta-action sequence, as shown in Figure 1. It indicates that the
6 meta-actions are quite diverse in the SUP-AD dataset. We also show the distribution of the length of
7 meta-actions per scene in Figure 2. Most scenes contain two or three meta-actions, and a few scenes
8 with complex driving strategies contain four or more meta-actions.

9 **Annotation of meta-actions.** The meta-action sequence for each driving scene is manually anno-
10 tated based on the actual driving strategy in the future frames. These meta-actions are designed to
11 encompass a complete driving strategy and are structured to be consistent with the future trajectory
12 of the ego vehicle. They can be divided into three primary classes:

- 13 1. **Speed-control actions.** Discerned from acceleration and braking signals within the ego
14 state data, these actions include These actions can be discerned from acceleration and brak-
15 ing signals within the ego state data. They include *speed up*, *slow down*, *slow down rapidly*,
16 *go straight slowly*, *go straight at a constant speed*, *stop*, *wait*, and *reverse*.
- 17 2. **Turning actions.** Deduced from steering wheel signals, these actions consist of *turn left*,
18 *turn right*, and *turn around*.
- 19 3. **Lane-control actions.** Encompassing lane selection decisions, these actions are derived
20 from a combination of steering wheel signals and either map or perception data. They
21 involve *change lane to the left*, *change lane to the right*, *shift slightly to the left*, and *shift*
22 *slightly to the right*.

23 A.2 Scenario Categories

24 The SUP-AD dataset is comprised of 1,000 video clips of driving scenarios. As illustrated in Fig-
25 ure 3, it encompasses a wide range of driving scenarios, spanning over 40 categories. Below are
26 explanations for some of the scenarios:

27 **AEB Data:** Automatic Emergency Braking (AEB) data.

28 **Road Construction:** A temporary work zone with caution signs, barriers, and construction equip-
29 ment ahead.

30 **Close-range Cut-ins:** A sudden intrusion into the lane of the ego vehicle by another vehicle.

31 **Roundabout:** A type of traffic intersection where vehicles travel in a continuous loop.

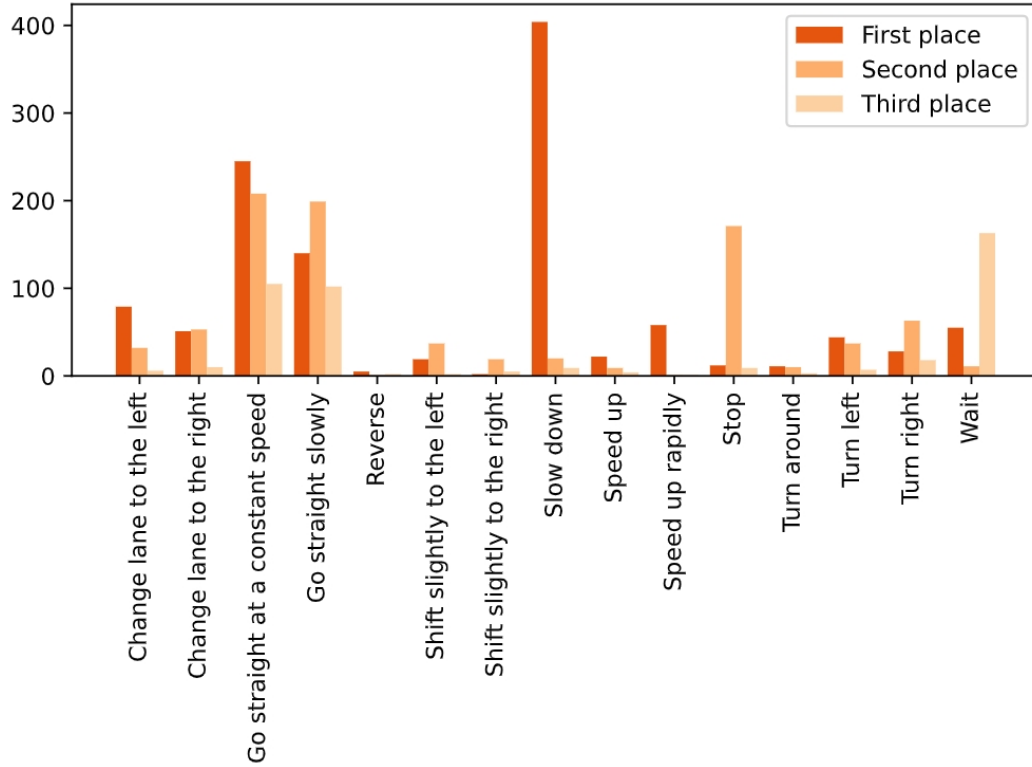


Figure 1: Distribution of each meta action being the first, second, and third place of the meta action sequence, respectively.

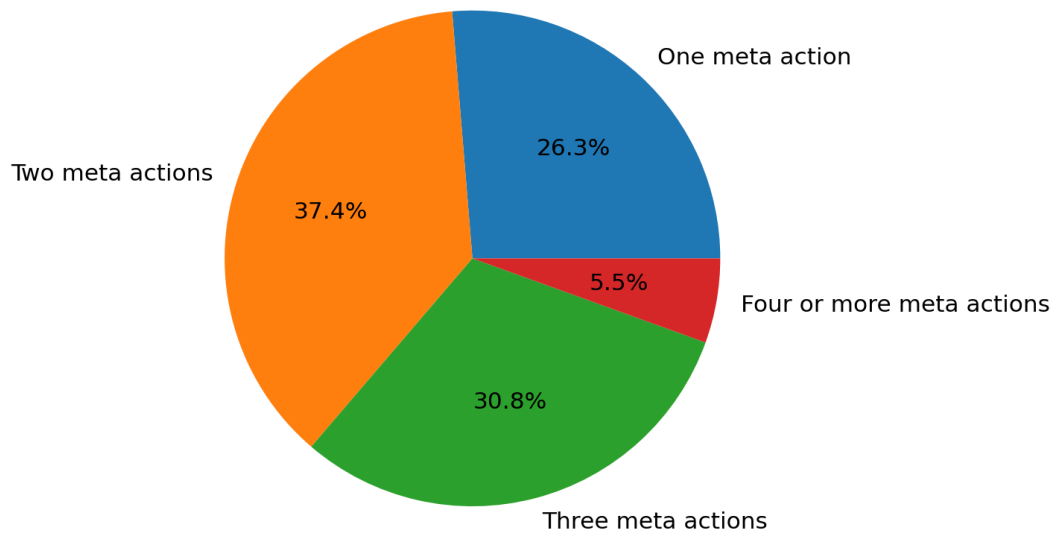


Figure 2: Distribution of the length of meta actions per scene.

³² **Animals Crossing Road:** Animals crossing the road in front of the ego vehicle.

³³ **Braking:** Brake is pressed by human driver of the ego vehicle.



Figure 3: **Diverse driving scenarios in the SUP-AD dataset.**

- 34 **Traffic Police Officers:** Traffic police officers managing and guiding traffic.
- 35 **Blocking Traffic Lights:** A massive vehicle obscuring the visibility of the traffic signal.
- 36 **Cutting into Other Vehicle:** Intruding into the lane of another vehicle ahead.
- 37 **Ramp:** A curved roadway that connects the main road to the branch road in highway.
- 38 **Debris on the Road:** Road with different kinds of debris.
- 39 **Narrow Roads:** Narrow roads that require cautious navigation.
- 40 **Pedestrians Popping Out:** Pedestrians popping out in front of the ego vehicle, requiring slowing
- 41 down or braking.

42 **People on Bus Posters:** Buses with posters, which may interfere the perception system.

43 **Merging into High Speed:** Driving from a low-speed road into a high-speed road, requiring speed-
 44 ing up.

45 **Barrier Gate:** Barrier gate that can be raised obstructing the road.

46 **Fallen Trees:** Fallen trees on the road, requiring cautious navigation to avoid potential hazards.

47 **Complex Environments:** Complex driving environments that requiring cautious navigation.

48 **Mixed Traffic:** A congested scenario where cars, pedestrians, and bicycles appear on the same or
 49 adjacent roadway.

50 **Crossing Rivers:** Crossing rivers by driving on the bridge.

51 **Screen:** Roads with screens on one side, which may interfere the perception system.

52 **Herds of Cattle and Sheep:** A rural road with herds of cattle and sheep, requiring careful driving
 53 to avoid causing distress to these animals.

54 **Vulnerable Road Users:** Road users which are more susceptible to injuries while using roads, such
 55 as pedestrians, cyclists, and motorcyclists.

56 **Road with Gallet:** A dusty road with gallet scattered across the surface.

57 The remaining scenario categories are: Motorcycles and Trikes, Intersection, People carrying Um-
 58 brella, Vehicles Carrying Cars, Vehicles Carrying Branches, Vehicles with Pipes, Strollers, Children,
 59 Tunnel, Down Ramp, Sidewalk Stalls, Rainy Day, Crossing Train Tracks, Unprotected U-turns,
 60 Snowfall, Large Vehicles Invading, Falling Leaves, Fireworks, Water Sprinklers, Potholes, Over-
 61 turned Motorcycles, Self-ignition and Fire, Kites, Agricultural Machinery.

62 A.3 Dataset Construction Pipeline

63 The dataset construction pipeline sequentially includes Long-tail Object/Challenging Scenario Min-
 64 ing, Keyframe Selection, Scene Annotation, and Verification.

65 **Long-tail Object Mining.** According to real-world road object distribution, we first define a list
 66 of long-tail object categories, such as weird-shaped vehicles, road debris, and animals crossing the
 67 road. Next, we mine these long-tail scenarios using a CLIP-based search engine, capable of mining
 68 driving data using language queries from a large collection of logs. Following that, we perform a
 69 manual inspection to filter out scenes inconsistent with the specified categories.

70 **Challenging Scenario Mining.** In addition to long-tail objects, we are also interested in challenging
 71 driving scenarios, where the driving strategy of the ego vehicle needs to be adapted according to the
 72 changing driving conditions. These scenarios are mined according to the variance of the recorded
 73 driving maneuvers.

74 **Keyframe Selection.** Each scene is a video clip, it is essential to identify the ‘keyframe’ to annotate.
 75 In most challenging scenarios, a keyframe is the moment before a significant change in speed or
 76 direction is required. We select this keyframe 0.5s to 1s earlier than the actual maneuver, based on
 77 comprehensive testing, to guarantee an optimal reaction time for decision-making. For scenes that
 78 do not involve changes in driving behavior, we select a frame that is relevant to the current driving
 79 scenario as the keyframe.

80 **Scene Annotation and Verification.** We employ a group of annotators to perform the scene anno-
 81 tation, including scene description, scene analysis, and planning, except for waypoints, which can
 82 be auto-labeled from the vehicle’s IMU recordings. To facilitate scene annotation, we make a video
 83 annotation tool with the following features: (1) the annotators can slide the progress bar back and
 84 forth to replay any part of a video; (2) while annotating a keyframe, the annotator can draw bound-
 85 ing boxes on the image together with language descriptions; (3) annotators can select from a list

86 of action and decision candidates while annotating driving plans. Each annotation is meticulously
87 verified by 3 annotators for accuracy and consistency, ensuring a reliable dataset for model training.

88 A.4 Annotation Examples

89 We provide more examples of annotation contents in Figure 4, 5, 6, 7, 8, and 9. The scenario
90 categories of these examples are overturned bicycles and motorcycles, herds of cattle and sheep,
91 collapsed trees, crossing rivers, barrier gate, and snowfall respectively.

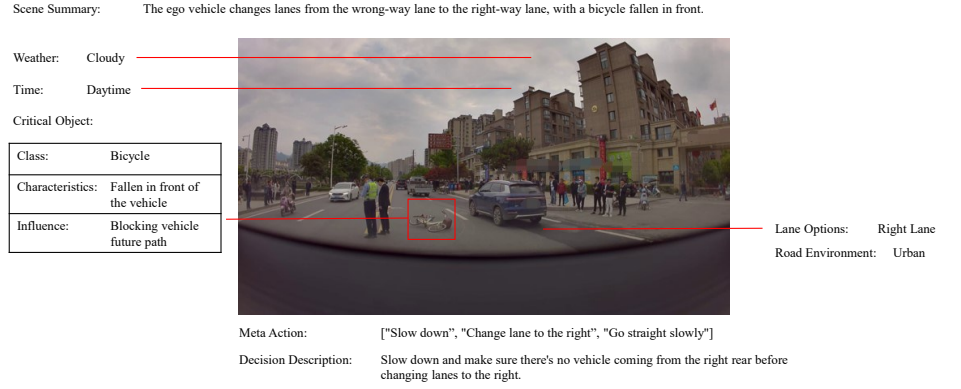


Figure 4: An example of overturned bicycles and motorcycles in the SUP-AD dataset. A bicycle has fallen in front of the ego vehicle, requiring the ego vehicle to change lanes.

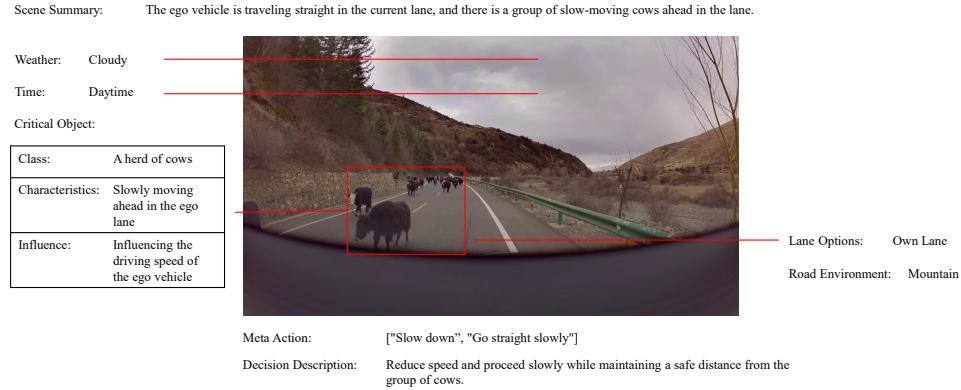


Figure 5: An example of herds of cattle and sheep in the SUP-AD dataset. A group of cattle move slowly in front of the ego vehicle, requiring the ego vehicle to proceed slowly and maintain a safe distance from the cattle.

Scene Summary: The ego vehicle is moving forward on the current road, and a tree suddenly falls towards the ego vehicle from the left front side.

Critical Object:

Class:	Tree
Characteristics:	Leaning towards our vehicle on the left front
Influence:	Blocking our vehicle from moving forward



Weather: Cloudy

Time: Daytime

Lane Options: Own Lane

Road Environment: National Road

Meta Action: ["Slow down rapidly", "Stop", "Wait"]

Decision Description: Immediately decelerate and come to a stop, wait for the fallen tree to be cleared before resuming driving.

Figure 6: An example of collapsed trees in the SUP-AD dataset. A tree suddenly falls towards the ego vehicle, requiring the ego vehicle to decelerate immediately.

Scene Summary: The ego vehicle travels at a constant speed along the current road towards a bridge ahead. The bridge width allows only a single vehicle to pass.

Critical Object:

Class:	Narrow bridge
Characteristics:	Passable width for only a single vehicle
Influence:	No stopping allowed



Weather: Cloudy

Time: Daytime

Lane Options: No Lane Marking

Road Environment: Narrow Bridge

Meta Action: ["Slow down", "Go straight slowly"]

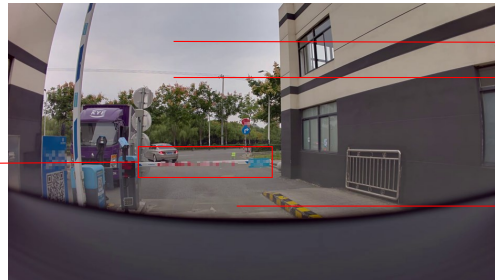
Decision Description: Brake and decelerate, drive slowly towards the bridge without stopping on it.

Figure 7: An example of crossing rivers in the SUP-AD dataset. The ego vehicle is going across a bridge of which width allows only a single vehicle to pass, requiring the ego vehicle to drive without stopping.

Scene Summary: The ego vehicle turns left towards the park entrance, a horizontal bar is blocking the entrance ahead.

Critical Object:

Class:	Crossbar
Characteristics:	At the entrance/exit ahead
Influence:	Blocking the vehicle's driving route



Weather: Cloudy

Time: Daytime

Lane Options: No Lane Marking

Road Environment: Park

Meta Action: ["Slow down", "Stop", "Wait"]

Decision Description: "Slow down and stop in front of the horizontal barrier, waiting for permission to continue."

Figure 8: An example of barrier gate in the SUP-AD dataset. A horizontal barrier blocks the entrance of a park, requiring the ego vehicle to stop and wait for permission to continue.



Figure 9: **An example of snowfall in the SUP-AD dataset.** Most of the road is covered by snow, requiring the ego vehicle to move forward cautiously by following the snow-free tire tracks.

92 B Evaluation Method

93 The ability of an autonomous driving system to accurately interpret driving scenes and make logical, suitable decisions is of paramount importance. As presented in this paper, the evaluation of
94 VLMs in autonomous driving concentrates on two primary components: the evaluation of scene
95 description/analysis and the evaluation of meta-actions.
96

97 B.1 Scene Description/Analysis Evaluation

98 In terms of scene description/analysis evaluation, the process of interpreting and articulating driving
99 scenes is subject to inherent subjectivity, as there are numerous valid ways to express similar
100 descriptions textually, which makes it difficult to effectively evaluate the scene description using
101 a fixed metric. To overcome this challenge, we utilize GPT-4 to evaluate the similarity between
102 the scene descriptions generated by the model and the manually annotated ground truth. Initially,
103 we prompt GPT-4 to extract individual pieces of information from each scene description. Subsequently,
104 we score and aggregate the results based on the matching status of each extracted piece of
105 information.

106 The ground truth labels for scene descriptions encompass both environment descriptions and event
107 summaries. Environmental condition description includes weather conditions, time conditions, road
108 environment, and lane conditions. Event summaries are the characteristics and influence of critical
109 objects. We employ GPT-4 to extract unique key information from both environment descriptions
110 and event summaries. The extracted information is then compared and quantified. Each matched
111 pair is assigned a score, which is estimated based on the extent of the matching, whether complete,
112 partial, or absent. Instances of hallucinated information incur a penalty, detracting from the overall
113 score. The aggregate of these scores constitutes the scene description score.

$$\text{Score} = \frac{1.0 \times n_{\text{matched}} + 0.5 \times n_{\text{partial}}}{n_{\text{gt}}} - \frac{0.25 \times n_{\text{hallucination}}}{n_{\text{gt}}} \quad (1)$$

114 The prompt for GPT-4 in evaluating scene descriptions is carefully designed, as shown in Table 1.
115 Initially, a role prompt is employed to establish as an intelligent and logical evaluator, possessing a
116 comprehensive understanding of appropriate driving styles. This is followed by specifying the input
117 format, which informs GPT-4 that its task involves comparing an output description with a ground

System Prompt:

You are a smart and logical evaluator with extensive driving experience, and you will try your best to give reasonable and logical evaluation result.

Input Prompt:

Given two driving scenario descriptions, one is the reference description, the other is the output description from a model. Please extract key information from the reference description and use it as comparison standards to compare the proximity of the output descriptions. Additionally, evaluate the completeness of each piece of information. Key information extracted can be classified into environmental conditions and critical events. Environmental condition information includes weather, time, road environment, and lane conditions. A critical event describes the event related to ego vehicle's driving decisions, which is usually given in the format of critical object + verb/adj + influence on ego vehicle. An example of a critical event is like this: A construction zone is blocking the ego lane and the right lane in front, so the car needs to go around to avoid it. Now I will give you the reference description and the output description.

The reference description:

Weather: Sunny. || Time: Day. || Road Environment: Urban. || Lane Options: Left Lane, Own Lane, Right Lane. || Ego Lane Position: Middle Lane.

Critical Events: There are police officers ahead inspecting passing vehicles. The police officer in front of your lane is signaling you to stop for inspection. There are also vehicles queuing in the left lane. The right lane is a merging lane and is currently separated, so you cannot change lanes to the right.

The output description:

Weather: Sunny. || Time: Day.

There is a vehicle driving in the distance in front of you. There are traffic police on the left and right, and the traffic police signal to stop.

Compare the two descriptions using extracted key information and output a similarities score on the scale from 0 to 1.0 and an explanation to describe the logic of calculating the similarity scale. Each matched key information scores 1.0, and each partially matched key information scores 0.5. Hallucinations will be penalized as well. If any key information exists in the output but not in the reference, a score of -0.25 will be assigned. The final score is the total score divided by the number of key information extracted in the reference description. For example, if the total score is 1.5, and the total number of key information in reference description is 3, then the final score would be $1.5/3 = 0.5$. Please output the detailed extraction and evaluation process.

Response:

Reference description key information:

1. Weather: Sunny (environmental conditions)
2. Time: Day (environmental conditions)
3. Road Environment: Urban (environmental conditions)
4. Lane Options: Left Lane, Own Lane, Right Lane (environmental conditions)
5. Ego Lane Position: Middle Lane (environmental conditions)
6. Police officers ahead inspecting vehicles (critical event)
7. The police officer in front of ego vehicle signalling to stop (critical event)
8. Vehicles queuing in the left lane (critical event)
9. Cannot change to the right lane as it's a merging lane and currently separated (critical event)

Output description key information:

1. Weather: Sunny.
2. Time: Day.
3. Vehicle driving in the distance in front of driver
4. Traffic police on the left and right
5. Traffic police signal to stop

By comparing the output to the reference:

- Key information 1, 2, 6, and 7 are matched, each for 1.0 points (weather and time, police officers and their signal to stop)
- Key information 3 in the output is not mentioned in reference, deduction of -0.25 points.

No scores for:

- Key information 3, 4, 5, 8, and 9.

Total score = $(1.0 + 1.0 + 1.0 + 1.0 - 0.25) / 9 = 0.417$.

Table 1: Evaluation process of scene description by GPT-4. Our proposed evaluation method is not only capable of extracting and evaluating information from structured scene descriptions but is also applicable to unstructured descriptions.

118 truth description. This comparison is based on the extraction and analysis of key information from
119 both descriptions. Lastly, the prompt outlines the criteria for scoring, as well as the format for the
120 evaluation output, ensuring a structured and systematic approach to the evaluation process.

121 B.2 Meta-action Evaluation

122 The evaluation process for the meta-action sequence must consider both the quantity and the sequen-
123 tial arrangement of the matched meta-actions. We employ dynamic programming to compare the
124 model's output and the annotated ground truth. Our dynamic programming approach is similar to
125 the method utilized in identifying the longest common subsequence, albeit with two supplementary
126 considerations.

127 The first consideration acknowledges the unequal weighting of different meta-actions. For instance,
128 certain meta actions such as “*Slow Down*”, “*Wait*”, and “*Go Straight Slowly*” exhibit a greater em-
129 phasis on attitude rather than action. The presence or absence of these actions from a meta-action

sequence does not alter the basic semantic essence of driving decisions but rather modifies the driving strategy to be either more assertive or more cautious. For example, a meta action sequence of “Slow Down → Stop → Wait” conveys a similar driving decision as a sequence with only the meta action “Stop”. Consequently, these sequences should not incur a penalty comparable to other meta actions such as “Turn Left” or “Change Lane to the Right”. Therefore, these are designated as “conservative actions”, and a reduced penalty is applied when they do not match during sequence evaluation.

The second consideration addresses the potential semantic equality among different meta-action sequences. For example, the sequences “Change Lane to the Left → Speed Up → Go Straight At a Constant Speed → Change Lane to the Right” and “Change Lane to the Left → Speed Up Rapidly → Go Straight At a Constant Speed → Change Lane to the Right” might both represent valid approaches to overtaking a slow-speed vehicle ahead. Recognizing that different meta-action sequences might convey similar meanings, we initially use GPT-4 to generate variant sequences that have comparable semantic meanings, in addition to the unique ground truth meta-action sequence, as shown in Table 2. In the subsequent sequence-matching phase of the evaluation, all these variations, together with the manually annotated ground truth, are taken into consideration. The highest-scoring matching is then adopted as the definitive score for the final decision evaluation.

Prompt:

Given a predefined set of action tokens and a sequence of reference driving actions, find all alternative sequences of actions that have similar meanings but choose slightly different action tokens from the predefined action tokens.

Predefined action tokens: [Speed up, Slow down, Speed up rapidly, Slow down rapidly, Go straight slowly, Go straight at a constant speed, Turn left, Turn right, Change lane to the left, Change lane to the right, Shift slightly to the left, Shift slightly to the right, Stop, Wait, Turn around, Reverse].

For example, if the reference driving action sequence is: Slow down → Shift slightly to the right → Go straight at a constant speed, alternative action sequences with similar meanings could be:

1. Slow down → Change lane to the right → Go straight at a constant speed;
2. Slow down rapidly → Shift slightly to the right → Go straight at a constant speed;
3. Slow down → Change lane to the right → Go straight slowly;
4. Slow down → Shift slightly to the right → Go straight slowly.

Now the reference action sequence is: Slow down → Shift slightly to the right → Go straight at a constant speed. What are the alternative action sequences?

Response:

Given the reference sequence: “Slow down → Shift slightly to the right → Go straight at a constant speed.” Alternative action sequences with similar meanings based on the provided action tokens are:

1. Slow down → Change lane to the right → Go straight at a constant speed.
2. Slow down rapidly → Shift slightly to the right → Go straight at a constant speed.
3. Slow down → Change lane to the right → Go straight slowly.
4. Slow down → Shift slightly to the right → Go straight slowly. You’ve provided this example in your question.

Based on the tokens and the context, these are indeed appropriate alternative sequences.

Table 2: Example of generating alternative action sequences by GPT-4.

The state of dynamic programming is saved in a 2D matrix, wherein each row corresponds to a meta action in the ground truth action sequence, and each column corresponds to a meta action in the model output action sequence, noted as $S^{r,c}$. The dynamic programming initiates recursive calculations beginning from the first meta action of both sequences. Each element of the 2D matrix encompasses the optimal total score at the current matching position, as well as the preceding matching condition that yielded the optimal matching. In our dynamic programming algorithm, three transition equations govern distinct cases: S_{missing} for missing matching, $S_{\text{redundant}}$ for redundant matching, and S_{matching} for successful matching. Successful matching occurs when the meta action is identical at the r^{th} position in the reference sequence and the c^{th} position in the model-generated sequence. In the case of missing matching, the meta action at the r^{th} position in the reference sequence is

unmatched, prompting a comparison with the $r - 1^{th}$ position in the reference sequence and the c^{th} position in the model-generated sequence. Conversely, redundant matching implies that the meta action at the c^{th} position in the model-generated sequence is unmatched, leading to further examination of the r^{th} position in the reference and the $c - 1^{th}$ position in the model-generated sequence. The transformation equations for these cases are as follows:

$$\begin{aligned} S_{\text{missing}}^{r,c} &= S^{r-1,c} - p_{\text{missing}}, \\ S_{\text{redundant}}^{r,c} &= S^{r,c-1} - p_{\text{redundant}}, \\ S_{\text{matching}}^{r,c} &= S^{r-1,c-1} + s_{\text{matching}}, \\ S^{r,c} &= \max(S_{\text{missing}}^{r,c}, S_{\text{redundant}}^{r,c}, S_{\text{matching}}^{r,c}), \end{aligned} \quad (2)$$

where $s_{\text{matching}} = 1.0$ represents the reward score after a successful matching. If an action considered missing or redundant is classified as a conservative action, the penalties p_{missing} and $p_{\text{redundant}}$ are quantified as half of s_{matching} , i.e., 0.5. Conversely, if an action is not conservative, both penalties are assigned the same magnitude as s_{matching} , i.e., 1.0. This approach is based on the premise that omitting a crucial meta action or inaccurately introducing a non-existent one equally hampers the effectiveness of the action sequence. The final score $Score_{\text{action}}$ should be divided by the length of the selected reference meta-action sequence, formulated as follow:

$$Score_{\text{action}} = \frac{S^{r,c}}{N_r} \quad (3)$$

C Co-tuning

To preserve the LLM’s generalization capabilities during the fine-tuning process, we employed co-tuning with several additional datasets. These include the Talk2Car [1], BDDX [2], Drama [3], SUTD [4], and LLAVA [5] datasets. For each dataset, we conducted random sampling in a 1:1 ratio corresponding to the data volume of the SUP-AD and nuScenes datasets. Following this co-tuning approach, we found that the scores on the SUP-AD dataset, under the evaluation metrics of Scene Description and Meta Action, remained virtually unchanged, simultaneously ensuring the preservation of the LLM’s original capabilities and its generalization capacity.

D DriveVLM-Dual Onboard Deployment

Base LLM	Avg.	MMMU 5%	SEEDBench 20%	RefCOCO 15%	SUP-AD 15%	Drivelm-QA 7.5%	Drivelm-Grounding 7.5%	Realworld-VQA 15%
MobileLLaMA1.4B [6]	0.457	0.331	0.59	0.421	0.520	0.686	0.735	0.501
Qwen-1.8B [7]	0.477	0.340	0.622	0.492	0.523	0.680	0.725	0.518
Gemma-2B [8]	0.439	0.345	0.571	0.330	0.510	0.680	0.721	0.507
MiniCPM-2.4B [9]	0.482	0.379	0.64	0.444	0.539	0.676	0.717	0.553
MobileLLaMA2.7B [6]	0.496	0.348	0.635	0.557	0.546	0.683	0.725	0.536
Phi3-3.8B [10]	0.538	0.435	0.688	0.608	0.604	0.697	0.743	0.592
Qwen-4B [7]	0.511	0.366	0.671	0.603	0.515	0.681	0.735	0.562
Qwen-4B*	0.529	0.373	0.684	0.624	0.596	0.699	0.738	0.553

Table 3: Performance of different LLMs on various datasets using the LLAVA-1.5 [11] architecture with ViT-L-336 [12] as the image encoder. Note that * indicates using SigLIP-L-384 [13] as the image encoder.

Base LLM Due to the limited memory and bandwidth of the vehicle’s hardware, we cannot use overly large LLMs to maintain real-time inference. Therefore, we chose models with fewer than 4 billion parameters. As shown in Table 3 and 4, our experiments revealed that on the Orin architecture, the ”wide and shallow” Qwen series (wider and fewer layers) models outperform ”narrow and deep” models (narrower and more layers) in inference speed.

LLM	Prompt Length(tokens)	Prefill latency (s)	Prefill (tok/s)	Decode (tok/s)	Output (tok/s)	Decode latency (s)	Model Size (GB)	Layer Num	Head Size	Vocab Size
Gemma-2B [8]	1063	0.95	1121	40.9	59	1.44	4.7	18	256	256000
Phi3-4k [10]	1045	1.3	797.3	49	59	1.2	7.2	32	96	32064
MobileLLaMa-2.7B [6]	1047	0.92	1134	61.7	59	0.96	5.0	32	80	32000
MobileLLaMa-1.4B [6]	1047	0.23	4634	117.4	59	0.5	2.5	24	128	32000
Qwen4B [7]	1078	0.57	1882	44.5	59	1.33	7.5	40	128	151936
Qwen1.8B [7]	1078	0.23	4709	79.6	59	0.74	3.7	24	128	151936

Table 4: Inference performance of different LLMs after quantization and deployment on an OrinX chip. The Qwen series achieved the best performance.

Backbone	Token Length	Method	Weighted Total	MMMU 5%	SEEDBench 20%	RefCOCO 15%	SUP-AD 15%	Drivelm-QA 7.5%	Drivelm-Grounding 7.5%	Realworld-VQA 15%	PointVQA 15%
ViT-L-336 [12]	-	-	0.421	0.368	0.657	0.502	0.553	0.688	0.742	0.541	0.583
ViT-L-336	-	Fixed 1x2 + CA-256	0.413	0.388	0.62	0.327	0.543	0.699	0.733	0.536	0.563
ViT-L-336	-	Fixed 2x2 + CA-256	0.405	0.376	0.618	0.306	0.518	0.692	0.731	0.523	0.594
ViT-L-448	-	Fixed 1x2 + CA-256	0.383	0.391	0.542	0.184	0.528	0.689	0.718	0.47	0.562
ViT-L-336	-	Fixed 1x2 + S2	0.368	0.383	0.511	0.198	0.469	0.689	0.718	0.469	0.514
ViT-L-336	-	Fixed 1x2 + S2	0.368	0.381	0.537	0.207	0.544	0.698	0.729	0.46	0.554
ViT-L-336	-	DM-4 + CA-256	0.409	0.377	0.61	0.277	0.544	0.698	0.726	0.525	0.554
ViT-L-336	-	DM-6 + CA-256	0.381	0.381	0.53	0.162	0.523	0.7	0.729	0.466	0.584
ViT-L-336	-	DM-4 + PT	0.426	0.376	0.653	0.557	0.585	0.7	0.743	0.54	0.598
ViT-L-336	-	DM-4 + SM + LT	0.413	0.403	0.617	0.293	0.55	0.699	0.737	0.527	0.532
ViT-L-336	-	DM-4 + SM + AAP + LT	0.406	0.384	0.61	0.273	0.518	0.68	0.726	0.52	0.533
ViT-L-336	-	DM-4 + SM + CD + LT	0.409	0.388	0.63	0.48	0.515	0.7	0.734	0.524	0.577
SigLIP-L-384 [13]	576	-	0.432	0.389	0.631	0.615	0.624	0.707	0.749	0.556	0.56
SigLIP-L-768	576	-	0.438	0.377	0.642	0.58	0.638	0.712	0.764	0.561	0.556
SigLIP-L-1152	576	-	0.436	0.377	0.637	0.595	0.628	0.715	0.763	0.571	0.564
SigLIP-L-384-768	576	PE	0.434	0.367	0.632	0.581	0.631	0.716	0.762	0.556	0.554
SigLIP-L-512-960	480	PE	0.442	0.369	0.64	0.557	0.65	0.719	0.762	0.579	0.568

Table 5: Performance of different methods for scaling ViT’s original input resolution to higher resolution. “CA” stands for cross-attention, “DM” is the abbreviation for Dynamic Max, “PT” indicates the use of patch end token, “SM” stands for Spatial Merge, “LT” indicates the use of line end token, “AAP” is the abbreviation for Adaptive Average Pooling, and “CD” stands for Convolutional Down-sampling. “PE” represents Position Embedding Interpolation. Among these methods, applying PE interpolation to SigLIP-L-384, transforming it to accept 768 resolution images as input, achieves a good trade-off between inference speed and performance.

Visual Encoder High-resolution images are essential for fine-grained understanding in autonomous driving. As shown in Table 5, compared to the basic ViT model used as a visual encoder, we explored several options, including different GridPatch strategies and PE (Position Embedding) interpolation. Ultimately, for real-time inference, we selected the simpler SigLIP-L-384 model with PE interpolation, achieving high resolution input through original 384 resolution PE interpolation and fine-tuning parameters with additional conv layers.

Visual Token Compression To address the increased computational load from high-resolution images, we implemented LDPNetv2 [6] to reduce the number of image tokens by 75% without compromising performance, as shown in Table 6. Additionally, we enhanced performance by replacing avg-pooling layer with convolutional layer in LDPNetv2.

Speculative Sampling Speculative Sampling is used to accelerate inference by preemptively generating likely outputs. This approach reduces the latency of generating predictions, achieving a significant speedup without substantial loss in accuracy. As shown in Table 7, we test two speculative sampling method Medusa and Eagle with our inference framework designed specifically for OrinX chip. Eagle achieved a 2.7x speedup in decode latency compared to Medusa’s 2.17x, making real-time vehicle deployment feasible.

Projector	Origin	Compressed	Output(ms)	Prefill(ms)	Avg.	MMMU	SEEDV2	BDD	Drivelm-QA	Drivelm-grounding	Realworld-VQA	RefCOCO	SUP-AD
MLP (Baseline)	576	576	666.60	707.93	56.27	37.90	64.00	53.90	67.60	71.70	55.30	44.40	53.90
LDPNetV2 [6]	576	576	661.70	707.42	50.53	37.67	56.71	53.36	68.90	73.20	48.30	21.29	54.70
Perceiver Resampler [14]	576	576	637.20	666.20	49.71	39.93	58.84	52.21	68.37	71.90	49.61	19.01	48.70
Pixel shuffle	576	256	610.70	655.92	52.02	40.42	60.20	55.19	68.72	71.52	51.44	28.23	48.40
LDPNetV2	576	256	605.32	652.79	54.73	38.93	62.19	54.82	68.57	72.78	50.59	34.32	59.30
Pixel shuffle	576	144	604.98	645.61	49.40	38.98	61.63	56.06	69.18	73.18	51.05	30.67	58.80
LDPNetV2	576	144	597.15	646.77	55.56	38.93	62.63	56.23	68.93	72.48	50.59	33.14	59.30
LDPNetV2	576	64	616.20	645.18	56.24	39.40	61.80	54.60	68.80	72.60	51.00	41.88	61.20

Table 6: Performance of different methods for visual token compression. Using LDPNetV2 to compress the original tokens to 75% of the original token count, achieves the best trade-off between performance and speed.

	Base + q4f16_l1	Eagle [15] + q4f16_l1	Medusa [16] + q4f16_ft	Eagle + q4f16_ft	Eagle + q4f16_ft + Shrink Vocab Size(1024)
Quant Type	q4f16_l1	q4f16_l1	q4f16_ft	q4f16_ft	q4f16_ft
Input Size	(384, 960)	(384, 960)	(384, 960)	(384, 960)	(384, 960)
Prefill Tokens	604	604	613	604	613
Output Tokens	37	39	41	41	41
Prefill Speed (tok/s)	1818	1793	2160.64	2230	2175.35
Decode Speed (tok/s)	109	172	232.34	295.04	518.3
Prefill Latency (s)	0.332	0.340	0.284	0.276	0.274
Decode Latency (s)	0.328	0.216	0.176	0.130	0.071
Acceleration ratio	1	1.57	2.17	2.7	4.33

Table 7: Performance of different speculative sampling method. “Shrink Vocab Size” means we reduce the vocabulary to the 1024 most frequently used words. “q4f16_l1” is a 4-bit quantization method using a 16-bit floating-point representation for efficient model compression, while “q4f16_ft” includes subsequent fine-tuning to enhance performance post-quantization.

E Qualitative Results

To further demonstrate the effectiveness and robustness of our DriveVLM, we provide additional visualization results in Figure 10, 11, 12, 13, and 14. In Figure 10, DriveVLM recognizes the slowly moving vehicle ahead and provides a driving decision to change lanes for overtaking. In Figures 11 and 12, DriveVLM accurately identifies the type of unconventional vehicles and a fallen tree, demonstrating its capability in recognizing long-tail objects. In Figure 13, the traffic police signaling to proceed with hand gestures has been accurately captured by DriveVLM. In Figure 14, DriveVLM successfully recognizes the road environment of a roundabout and generates a planned trajectory with a curved path.

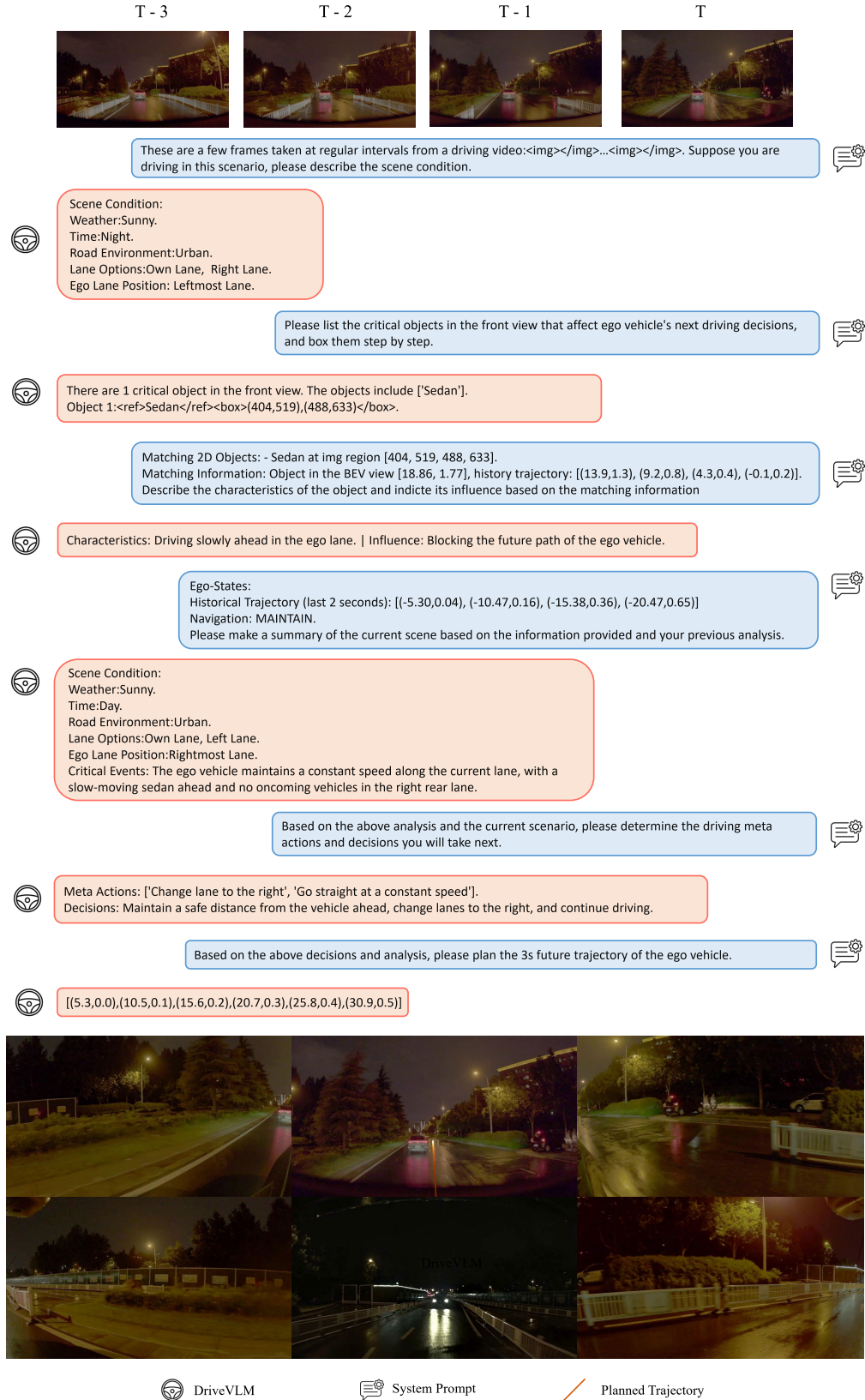


Figure 10: **Visualization of DriveVLM’s output.** DriveVLM recognizes the slowly moving vehicle ahead and provides a driving decision for changing lanes to overtake.



Figure 11: **Visualization of DriveVLM's output.** DriveVLM has identified the type of the unconventional vehicle ahead.



Figure 12: **Visualization of DriveVLM's output.** DriveVLM precisely detect the fallen tree and its position, subsequently planning an appropriate detour trajectory.



Figure 13: **Visualization of DriveVLM's output.** The traffic police signaling to proceed with hand gestures has been accurately captured by DriveVLM.

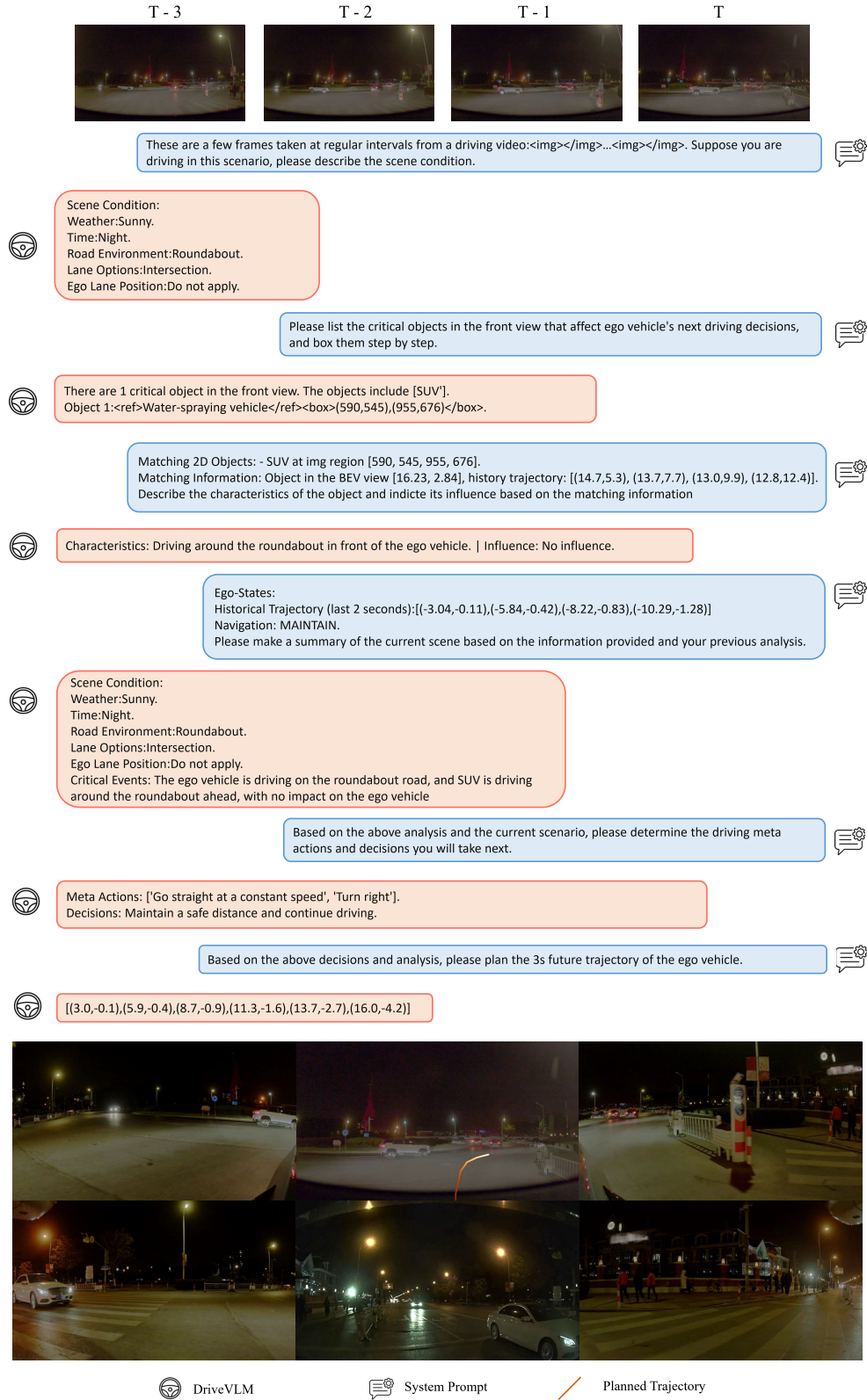


Figure 14: **Visualization of DriveVLM’s output.** DriveVLM successfully recognizes the road environment of a roundabout and generates a planned trajectory with a curved path.

References

- [1] T. Deruyttere, S. Vandenhende, D. Grujicic, L. Van Gool, and M.-F. Moens. Talk2car: Taking control of your self-driving car. *arXiv preprint arXiv:1909.10838*, 2019.
- [2] J. Kim, A. Rohrbach, T. Darrell, J. Canny, and Z. Akata. Textual explanations for self-driving vehicles. In *Proceedings of the European conference on computer vision (ECCV)*, pages 563–578, 2018.
- [3] S. Malla, C. Choi, I. Dwivedi, J. H. Choi, and J. 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.
- [4] L. Xu, H. Huang, and J. Liu. Sutd-trafficqa: A question answering benchmark and an efficient network for video reasoning over traffic events. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9878–9888, 2021.
- [5] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.
- [6] X. Chu, L. Qiao, X. Lin, S. Xu, Y. Yang, Y. Hu, F. Wei, X. Zhang, B. Zhang, X. Wei, et al. Mobilevlm: A fast, reproducible and strong vision language assistant for mobile devices. *arXiv preprint arXiv:2312.16886*, 2023.
- [7] Q. Team. Introducing qwen1.5, February 2024. URL <https://qwenlm.github.io/blog/qwen1.5/>.
- [8] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.
- [9] S. Hu, Y. Tu, X. Han, C. He, G. Cui, X. Long, Z. Zheng, Y. Fang, Y. Huang, W. Zhao, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024.
- [10] M. Abdin, S. A. Jacobs, A. A. Awan, J. Aneja, A. Awadallah, H. Awadalla, N. Bach, A. Bahree, A. Bakhtiari, H. Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.
- [11] H. Liu, C. Li, Y. Li, and Y. J. Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306, 2024.
- [12] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [13] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023.
- [14] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.
- [15] Y. Li, F. Wei, C. Zhang, and H. Zhang. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*, 2024.
- [16] T. Cai, Y. Li, Z. Geng, H. Peng, and T. Dao. Medusa: Simple framework for accelerating llm generation with multiple decoding heads, 2023.