ReAL-AD: Towards Human-Like Reasoning in End-to-End Autonomous Driving Supplementary Materials

Yuhang Lu[1], **Jiadong Tu**[1], **Yuexin Ma**[1,†], **Xinge Zhu**[2,†]
[1] ShanghaiTech University
[2] The Chinese University of Hong Kong
{luyh2, tujd2023, mayuexin}@shanghaitech.edu.cn, zhuxinge123@gmail.com

In the appendix, we first present additional model details, followed by analytical experiments to further demonstrate our model's effectiveness, and finally we provide visualization results.

A. More details about method

In this section, we provide more details of our proposed ReAL-AD framework, covering the Strategic Reasoning Injector (SRI) module, the Tactical Reasoning Integrator (TRI) module, the Hierarchical Trajectory Decoder (HTD) module, and model adaptations for closed-loop evaluation.

A.1. More details of SRI module

Driving Strategy Prompt. The prompt used to generate driving strategy text is as follow:

Strategy Prompt

There is an image from front camera of the car. Suppose you are driving in this scenario, based on the scene condition and critical objects or traffic signs in the scene, make legal, safe and comfortable driving decisions (may include which traffic participants to pay attention to, which traffic rules to pay attention to, etc.). Answer should be in one paragraph and the format is as follows:
`'When driving in the current scenario, the driving decision-making thinking process is:...'.`

Details about ground truth trajectory encoder. In the SRI module, we encode the ground truth trajectory into planning features to ensure consistency between strategy semantics and planning dynamics. The pseudo-code for this encoder is presented in Algorithm 1.

Algorithm 1 Ground-truth trajectory encoder

Input: Ground truth trajectory gt_{traj} of shape (B, N, T, D)
Hyperparameters: Number of subgraph layers L, Hidden dimension H
Output: Extracted trajectory features F_{gt}
Initialize feature representation $F \leftarrow gt_{traj}$
for $i = 1$ to L **do**
  $F \leftarrow \text{MLP}_i(F)$ ▷ Apply MLP to each point feature
  $F_{\text{max}} \leftarrow \max(F, \text{axis} = -2)$ ▷ Max pooling along sequence length
  $F_{\text{max\_expanded}} \leftarrow \text{Expand}(F_{\text{max}}, \text{axis} = -2, \text{size} = T)$
  $F \leftarrow \text{Concat}(F, F_{\text{max\_expanded}}, \text{axis} = -1)$ ▷ Feature concatenation
end for
$F_{\text{gt}} \leftarrow \max(F, \text{axis} = -2)$ ▷ Final max pooling for global trajectory features
return F_{gt}

Beyond ensuring consistency, aligning the encoded ground truth trajectory with the loss of the driving strategy features provides additional benefits. In rare instances where the driving strategy inferred by the VLM conflicts with the generated driving command, the GT trajectory serves as a common supervisory signal for both. Specifically, the loss between the driving strategy features and the GT trajectory helps refine the strategy representation, while the generated driving command, once injected into the decoder, is also supervised by the correct trajectory at both coarse and fine-grained levels. Since both the strategy features and the command share this trajectory-based supervision, the model benefits from a unified correction mechanism, effectively mitigating the impact of inconsistencies in the VLM's output.

Verification of VLM output strategy text. Due to differences in command-following capabilities among VLMs,

[0]† Corresponding authors.

they may not always adhere to the specified output format, even when explicitly instructed. To ensure the reliability of VLM-generated strategy text, we rigorously verify that the response begins with "*When driving in the current scenario...*" as specified in the prompt. If it does not, we regenerate the response.

A.2. More details of TRI module

Driving Command Prompt. To generate tactical command, we design the prompt as Fig. 1 shows.

Verification of VLM output driving command. Similar to strategy text, we use regular expressions to extract structures such as "*Direction Control:*" and three other similar categories from the text, identifying the word following each as the command reply. Additionally, each category has a predefined set of valid responses. If the VLM output does not match any of the expected answers within its respective category or if the structure is missing, the output is regenerated.

A.3. More details of HTD module.

Coarse trajectory ground truth generation. We convert the fine trajectory ground truth into a coarse trajectory through Bézier curve interpolation [2] and strategic control point sampling. This process consists of four key stages.

First, we transform the offset coordinates $\{\Delta\mathbf{P}_i\}_{i=1}^{6} \in \mathbb{R}^{6\times 2}$ into absolute coordinates via cumulative summation:

$$\mathbf{P}_i^{abs} = \begin{cases} \Delta\mathbf{P}_1, & i = 1 \\ \mathbf{P}_{i-1}^{abs} + \Delta\mathbf{P}_i, & 2 \le i \le 6 \end{cases} \tag{1}$$

This step ensures spatial continuity, which is crucial for the subsequent interpolation.

Next, we extract four key control points from $\{\mathbf{P}_i^{abs}\}$ to effectively capture kinematic patterns while maintaining smoothness. Specifically, we select $\mathbf{P}_1^{abs}$ as the starting point, $\mathbf{P}_3^{abs}$ and $\mathbf{P}_4^{abs}$ as intermediate points representing motion inflection regions, and $\mathbf{P}_6^{abs}$ as the endpoint.

To obtain a smoothed trajectory, we then apply cubic Bézier interpolation using these control points. The trajectory $\{\widetilde{\mathbf{P}}_i^{abs}\} \in \mathbb{R}^{6\times 2}$ is computed as:

$$\mathbf{B}(t) = \sum_{k=0}^{3} \mathbf{C}_{k+1} \binom{3}{k} t^k (1-t)^{3-k}, \quad t \in [0,1] \tag{2}$$

where we uniformly sample at $t = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ to maintain temporal consistency with the original trajectory.

Finally, we convert the interpolated absolute coordinates back to the offset form:

$$\Delta\widetilde{\mathbf{P}}_i = \begin{cases} \widetilde{\mathbf{P}}_1^{abs}, & i = 1 \\ \widetilde{\mathbf{P}}_i^{abs} - \widetilde{\mathbf{P}}_{i-1}^{abs}, & 2 \le i \le 6 \end{cases} \tag{3}$$

The resulting coarse trajectory $\{\Delta\widetilde{\mathbf{P}}_i\}$ preserves the global motion trends and serves as supervision for the network's predicted trajectory.

Distribution Encoder. In this module, we leverage Distribution Encoder to map decoder input into latent trajectory space. The operation is as follow:

Algorithm 2 Distribution Encoder

Input: Input tensor s_t, consisting of concatenated ego-vehicle features, corresponding command features, and optionally coarse latent code, with shape (B, T, D).
Hyperparameters: Latent dimension L, Minimum log variance min_log_sigma, Maximum log variance max_log_sigma
Output: Mean μ and log standard deviation $\log\sigma$
Initialize compression dimension $C \leftarrow D/2$
Initialize encoder E with three convolutional layers:
    Conv1 : $(D \to 2D, 1\times 1)$, ReLU
    Conv2 : $(2D \to 2D, 1\times 1)$, ReLU
    Conv3 : $(2D \to C, 1\times 1)$
Initialize final convolutional layer LastConv:
    Adaptive average pooling $(T \to 1)$
    Conv : $(C \to 2L, 1\times 1)$
Encoding Process:
$s_t \leftarrow$ Permute$(s_t, (0,2,1))$
$F \leftarrow E(s_t)$
$mu\_log\_sigma \leftarrow$ LastConv(F)
$mu\_log\_sigma \leftarrow$ Permute$(mu_log_sigma, (0,2,1))$
Extract Distribution Parameters:
$\mu \leftarrow mu\_log\_sigma[:,:,:L]$
$\log\sigma \leftarrow mu\_log\_sigma[:,:,L:]$
$\log\sigma \leftarrow$ Clamp$(\log\sigma$, min_log_sigma, max_log_sigma$)$
return $\mu, \log\sigma$

A.4. Model adaptation for closed-loop evaluation

For closed-loop evaluation, we adapted the model to improve processing speed, particularly for the slow VLM network. To reduce VLM usage frequency and increase FPS, we implemented a memory buffer mechanism that stores environmental features along with the corresponding VLM policy text and driving instruction features from the open-loop evaluation dataset. During closed-loop evaluation, when encountering a new scene, the system first extracts environmental features and computes their similarity with those in the buffer. It then identifies the most similar stored frame based on environmental feature similarity and reuses its VLM policy text and driving instruction features for the current frame, ensuring efficient decision-making.

B. More Analysis

In this section, we will provide more analytical experiments on our proposed modules. All experimental settings are

Command Prompt

There is an image from the front camera of the car. Suppose you are driving in this scenario, please analyze the following input image and determine the ego-vehicle's planning commands based on the current road conditions. Strictly adhere to the predefined command types listed below and return the results in the required format. The output must only include the specified options and should not contain any additional content.

Command Types

Primary Direction Control

- LEFT_TURN: Make a left turn at the upcoming intersection or turn
- RIGHT_TURN: Make a right turn at the upcoming intersection or turn
- CONTINUE_STRAIGHT: Proceed straight without changing direction. Maintain the current lane unless a lane change is necessary for traffic flow

Lane Positioning

- KEEP_LANE: Maintain the current lane without changing
- CHANGE_LANE_LEFT: Move to the adjacent lane on the left to facilitate a left turn or overtake slower traffic
- CHANGE_LANE_RIGHT: Move to the adjacent lane on the right to facilitate a right turn or overtake slower traffic

Speed Regulation

- ACCELERATE: Increase the vehicle's speed
- DECELERATE: Decrease the vehicle's speed
- MAINTAIN_SPEED: Keep the current speed constant

Emergency Control

- EMERGENCY_BRAKE: Apply brakes immediately to avoid a collision or hazard
- PARK: Bring the vehicle to a complete stop and park
- NO_ACTION: No emergency control action is required in the current scenario

Output Format

```
Direction Control: <Direction Control>
Lane Management: <Lane Management>
Speed Control: <Speed Control>
Emergency Control: <Emergency Control>
```

Figure 1. Command prompt structure for tactical decision-making.

consistent with those in the ablation study of main paper.

B.1. Discussion on design of the adapter in SRI module.

In the SRI module, the adapter aligns encoded strategy textual features with perceptual features. While our main implementation uses an MLP, we also experimented with a self-attention mechanism. However, Tab. 1 shows no significant performance difference. This suggests that the MLP is already sufficient for feature adaptation, as the alignment task does not heavily rely on long-range dependencies. Additionally, the MLP is more computationally efficient, making it a practical choice.

B.2. Discussion on similarity loss selection in SRI module.

To investigate the impact of different similarity loss functions in the SRI module, we conduct an ablation study comparing Cosine Similarity Loss (our default choice) with Mean Squared Error (MSE) Loss and Kullback-Leibler (KL) Divergence Loss. The results, as shown in Tab. 2, indicate that MSE Loss performs the worst, likely due to its sensitivity to magnitude differences, which can lead to instability when aligning high-dimensional feature spaces. Meanwhile, KL Divergence Loss and Cosine Similarity Loss achieve comparable performance, suggesting that capturing relative distribution differences (as in KL Loss) or directional alignment (as in Cosine Loss) is more effective

Sim Loss	L2(m) ↓				Collision Rate (%) ↓			
	1s	2s	3s	Avg.	1s	2s	3s	Avg.
SelfAttn	0.43	0.87	1.40	0.90	0.08	0.17	0.24	0.16
MLP	0.44	0.86	1.36	0.89	0.08	0.16	0.26	0.17

Table 1. Ablation study on different adapter designs in the SRI module.

Loss Function	L2(m) ↓				Collision Rate (%) ↓			
	1s	2s	3s	Avg.	1s	2s	3s	Avg.
None	0.45	0.88	1.43	0.92	0.09	0.16	0.27	0.18
MSE Loss	0.47	0.89	1.45	0.93	0.11	0.16	0.32	0.20
KL Loss	0.43	0.88	1.35	0.89	0.07	0.19	0.28	0.18
CosSim Loss	0.44	0.86	1.36	0.89	0.08	0.16	0.26	0.17

Table 2. Ablation study to different similarity loss functions in the SRI module.

Model	L2(m) ↓				Collision Rate (%) ↓			
	1s	2s	3s	Avg.	1s	2s	3s	Avg.
one-hot form	0.52	0.93	1.45	0.97	**0.08**	**0.14**	**0.27**	**0.16**
probabilistic form	**0.42**	**0.83**	**1.32**	**0.86**	0.11	0.20	0.28	0.20

Table 3. Ablation study for different forms of tactical driving commands.

than direct magnitude-based regression.

These findings reinforce that the specific choice of loss function is not the primary focus of our design. Instead, the core contribution lies in the conceptual framework of aligning these two features effectively, ensuring consistency between strategy semantics and planning dynamics.

B.3. Discussion on different forms of tactical driving commands.

As shown in Tab.3, we modify the prompt input to the VLM to generate tactical commands in a probabilistic form (see Fig.2) and encode them using a MLP, while keeping all other settings unchanged. However, this adjustment results in a performance decline. The probabilistic representation introduces uncertainty, making it more challenging for the model to produce clear, decisive predictions. In contrast, the one-hot format offers more direct supervision, leading to improved performance.

B.4. Discussion on camera input of VLMs.

Our current design focuses on front-view reasoning to balance computational efficiency and performance, as strategic and tactical decisions (e.g., lane changes) primarily depend on the front environment. We acknowledge, however, that multi-view input can be beneficial in special cases. To explore this, we use VAD and Qwen-VL with multi-view input but get limited improvement, suggesting that most decisions can be accurately made using the front view alone. But multi-view setting is truly deserved to further explore.

Model	Avg. L2 (m)	Avg. Col. Rate (%)
Ours	0.53	0.17
Ours with multi-view input	0.51	0.16

Table 4. Ablation study for different number of image input of VLM.

C. Visualization

In this section, we present comprehensive visualization results of our approach.

C.1. Qualitative results.

We use VAD [1] as the baseline and select MiniCPM-Llama3-2.5V [3] as the VLM, then perform a qualitative analysis of our proposed framework, as illustrated in Fig. 3, 4, 5 and 6. In Fig. 3, the baseline method failed to recognize the intention to change lanes, whereas our method accurately captured it and executed a safe lane change. In Fig. 4, when a pedestrian suddenly appeared, the baseline method responded too slowly and failed to perform an emergency avoidance maneuver, while our method successfully did so. In Fig. 5 and 6, the baseline method did not recognize the turning intention and missed the necessary turns, whereas our method correctly executed them. These results demonstrate that incorporating human reasoning into the end-to-end autonomous driving system enhances both the accuracy and safety of planning.

C.2. Demonstration of enhanced interpretability.

Our proposed module enhances the interpretability of end-to-end trajectory prediction by incorporating human-like reasoning. Taking the two challenging scenarios in Fig. 7 as examples, we observe that the generated driving strategies align with the final trajectory predictions, reinforcing interpretability. In the left scenario, the model explicitly suggests a lane change to continue driving, which matches the command-level output. In the right scenario, the strategy advises slowing down and monitoring pedestrians, while the command recommends emergency braking—both contributing to safe trajectory planning. This alignment between intermediate reasoning and final predictions makes the decision-making process more transparent and interpretable.

References

[1] Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. In *ICCV*, pages 8340–8350, 2023. 4

[2] Tim A Pastva. *Bezier curve fitting*. PhD thesis, Monterey, California. Naval Postgraduate School, 1998. 2

[3] Tianyu Yu, Haoye Zhang, Yuan Yao, Yunkai Dang, Da Chen, Xiaoman Lu, Ganqu Cui, Taiwen He, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Rlaif-v: Aligning mllms through open-source ai feedback for super gpt-4v trustworthiness. *arXiv preprint arXiv:2405.17220*, 2024. 4

Probability-based Command Prompt

There is an image from the front camera of the car. Suppose you are driving in this scenario, please analyze the following input image and determine the ego-vehicle's planning commands based on the current road conditions. **Strictly adhere to the predefined command types listed below and return the results in the required format. The output must **only** include the specified options and should **not** contain any additional content.**

Important:

1. You must provide a probability distribution for each command category, ensuring that the probabilities for each category sum up to 1.
2. Use format strictly as shown below without any extra text or explanations.
3. Refer to the provided examples to guide your output format and content.

Command Types

Emergency Control

- **EMERGENCY_BRAKE**: Immediately apply emergency brakes to avoid a collision or respond to a sudden hazard.
- **PARK**: Bring the vehicle to a complete stop and park, suitable for situations requiring immediate cessation.
- **NO_ACTION**: No emergency control action is required in the current scenario, and the vehicle continues normal operation.

Primary Direction Control

- **LEFT_TURN**: Make a left turn at the upcoming intersection or turn into another street.
- **RIGHT_TURN**: Make a right turn at the upcoming intersection or turn into a parking area or another street.
- **CONTINUE_STRAIGHT**: Proceed straight along the current route without changing direction.

Lane Management

- **KEEP_LANE**: Maintain the current lane without changing, suitable for smooth traffic flow within the lane.
- **CHANGE_LANE_LEFT**: Move to the adjacent lane on the left, typically for overtaking or preparing for a left turn.
- **CHANGE_LANE_RIGHT**: Move to the adjacent lane on the right, typically for overtaking or preparing for a right turn.

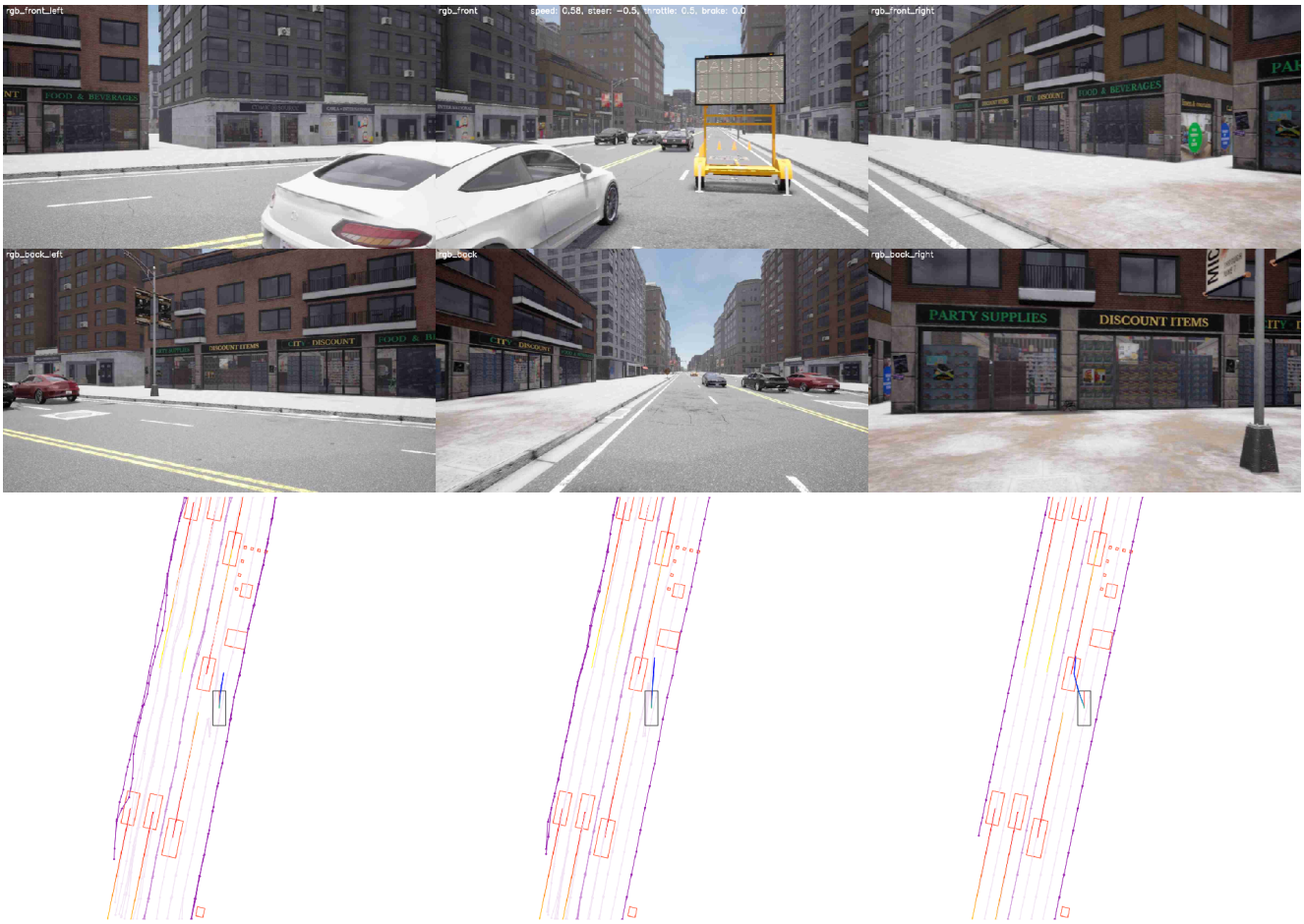
Speed Control

- **ACCELERATE**: Increase the vehicle's speed, suitable for smooth traffic flow ahead or when overtaking.
- **DECELERATE**: Decrease the vehicle's speed, suitable for congested traffic, reduced speed limits, or obstacles ahead.
- **MAINTAIN_SPEED**: Keep the current speed constant, suitable for stable traffic conditions or when no speed adjustment is needed.

Output Format

```
{
  "Emergency Control": {
    "EMERGENCY_BRAKE": <probability>,
    "PARK": <probability>,
    "NO_ACTION": <probability>
  },
  "Primary Direction Control": {
    "LEFT_TURN": <probability>,
    "RIGHT_TURN": <probability>,
    "CONTINUE_STRAIGHT": <probability>
  },
  "Lane Management": {
    "KEEP_LANE": <probability>,
    "CHANGE_LANE_LEFT": <probability>,
    "CHANGE_LANE_RIGHT": <probability>
  },
  "Speed Control": {
    "ACCELERATE": <probability>,
    "DECELERATE": <probability>,
    "MAINTAIN_SPEED": <probability>,
    "STOP": <probability>
  }
}
```

Figure 2. Command prompt structure for tactical decision-making with probability-based outputs.



VAD Ours GT

Figure 3. Visualization of our method, depicting a scenario where an error is detected in the current lane while driving, requiring a lane change to proceed.

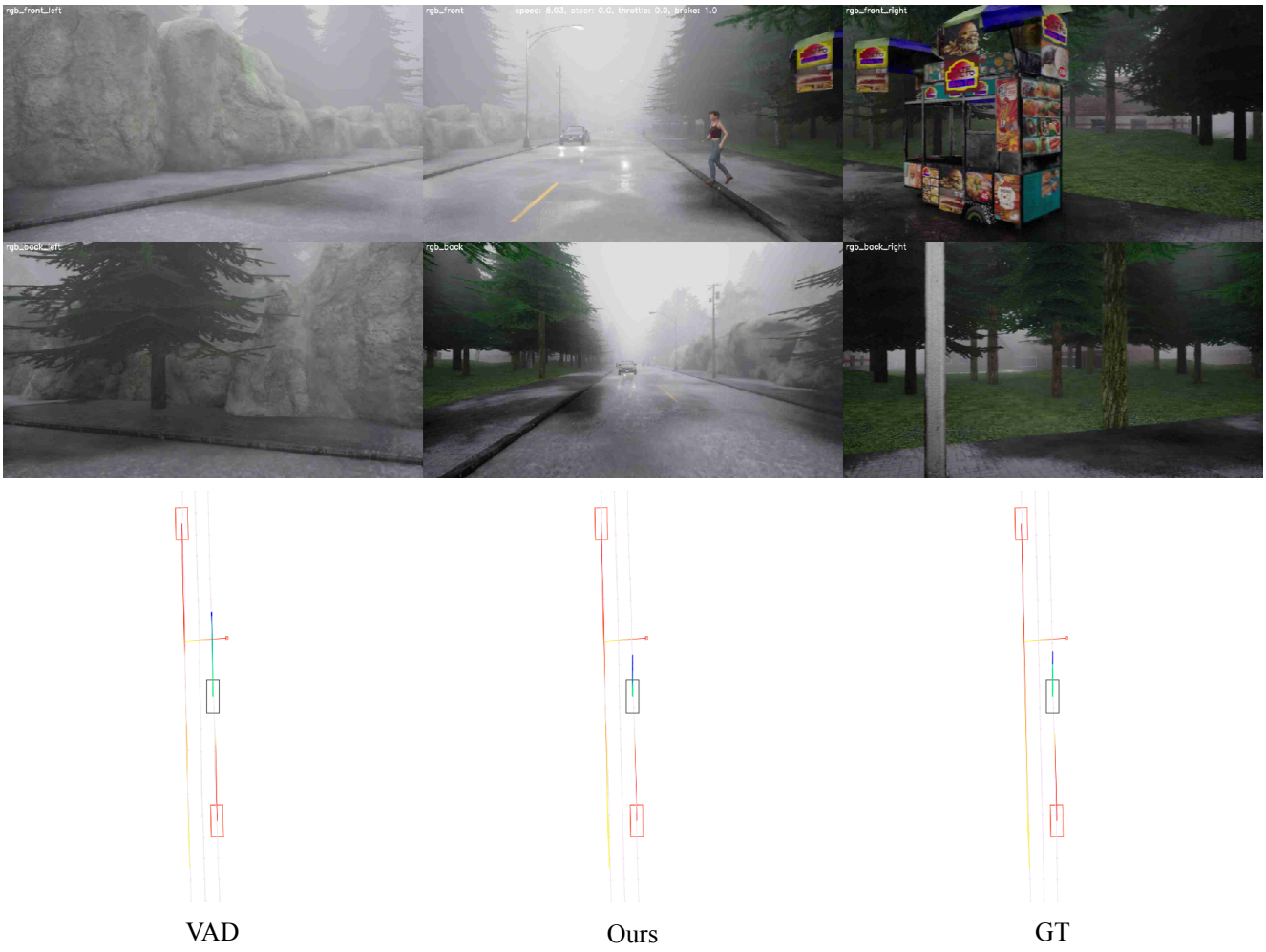


Figure 4. Visualization of our methods, depicting a scenario where a pedestrian suddenly appears in front of you and you need to make emergency avoidance while driving.

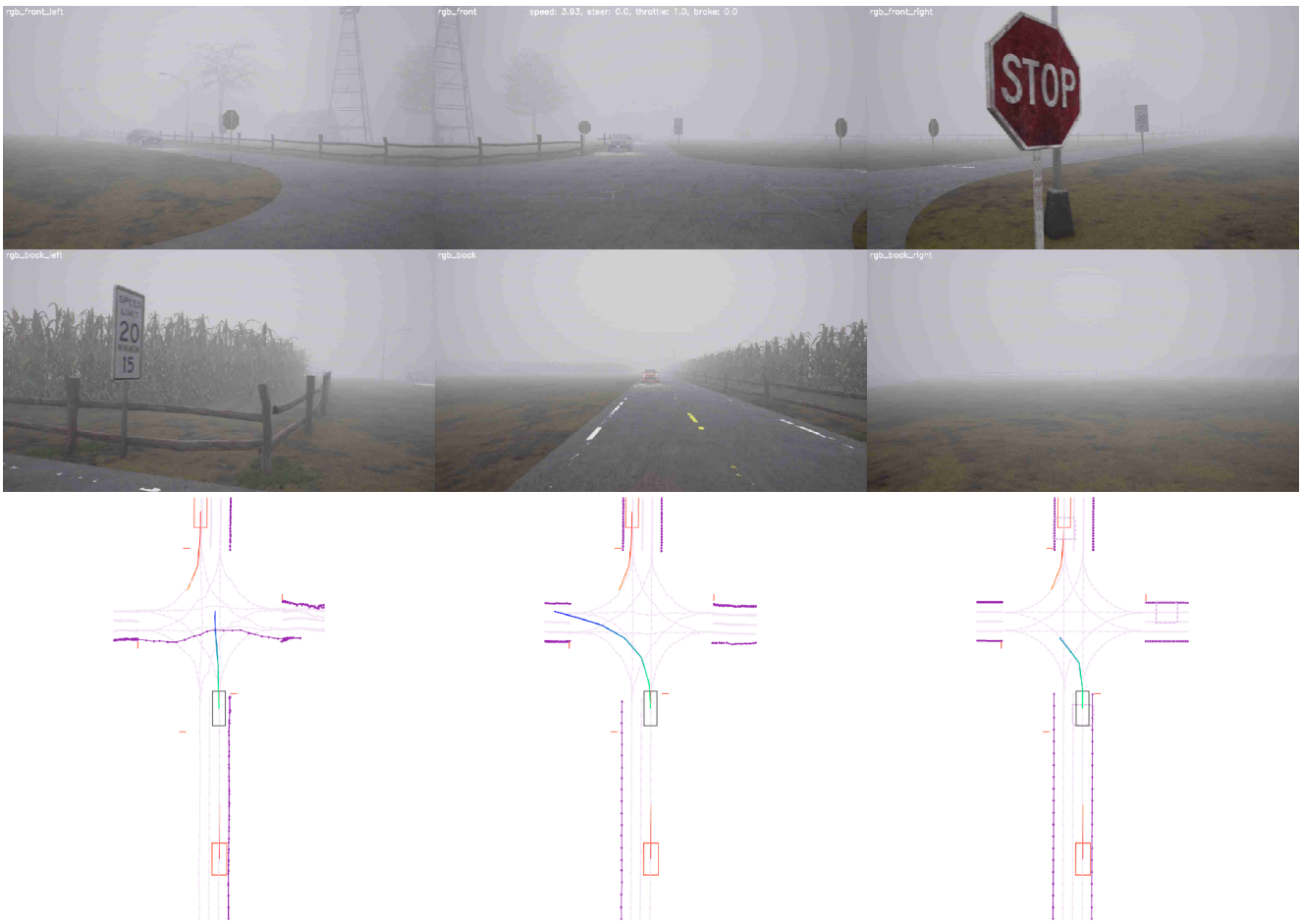


Figure 5. Visualization of our methods, depicting a turn without a signal.

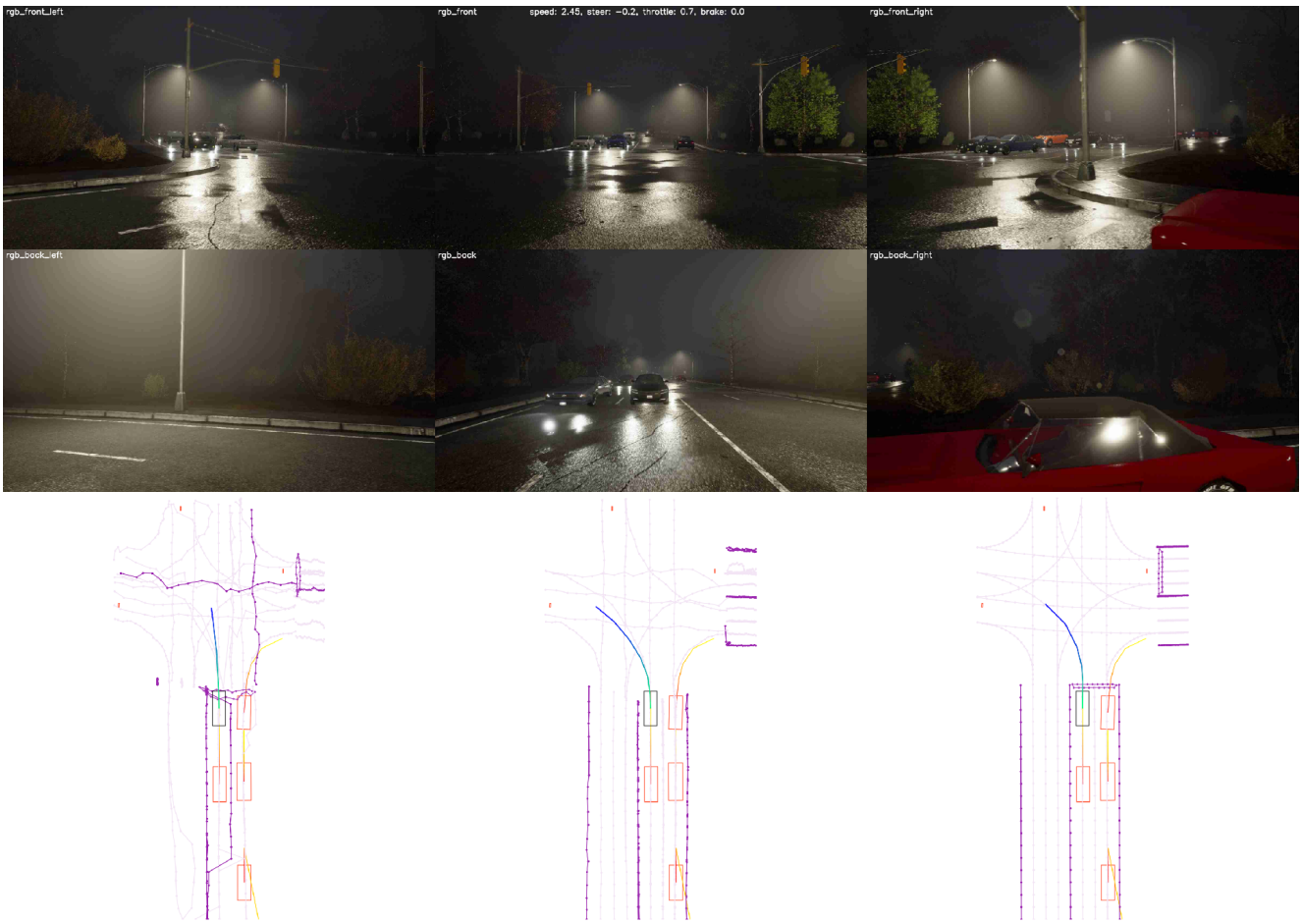


Figure 6. Visualization of our methods, depicting a turn with a signal.

VLM-generated driving strategy

First, observe and acknowledge the "Caution" sign, which indicates that there may be road construction or hazards ahead. This may require slowing down to ensure safety.

Next, consider traffic flow and vehicle positioning; if you are unable to continue in your current lane, you may want to consider changing lanes.

Maintain a safe following distance and be vigilant for any further instructions or changes in traffic conditions.

(Driving Strategy)

VLM-generated driving strategy

First, reduce speed due to the foggy conditions which impair visibility. Keep a safe distance from the car ahead to allow for extra reaction time.

Pay attention to pedestrians, such as the person crossing the street, and be prepared to stop if necessary.

Avoid sudden movements or lane changes that might startle other road users.

(Driving Strategy)

VLM-generated reactive-level commands

Direction Control: CONTINUE_STRAIGHT

Emergency Control: NO_ACTION

(Driving Decision)

VLM-generated reactive-level commands

Direction Control: CONTINUE_STRAIGHT

Emergency Control: EMERGENCY_BRAKE

(Driving Decision)

VLM-generated regulatory-level commands

Lane Management: CHANGE_LANE_LEFT

Speed Control: DECELERATE

(Driving Operation)

VLM-generated regulatory-level commands

Lane Management: KEEP_LANE

Speed Control: DECELERATE

(Driving Operation)

Figure 7. Visualization of VLM-generated driving strategies and tacical commands, demonstrating their alignment with final planning.