*d*VLM-AD: Enhance Diffusion Vision-Language-Model for Driving via Controllable Reasoning

Supplementary Material

A. Overview

Our supplementary material includes the following sections:

1. **Section** B: Data Construction Details. Details for annotations generation.
2. **Section** C: Evaluation Metric. Details of our evaluation settings and proposed consistency metrics.
3. **Section** D: Training Details. Details of our training paradigms.
4. **Section** E: Demonstration. We include more examples of our *d*VLM-AD compared with auto-regressive VLMs.
5. **Section** F: Prompts. We show all the prompts we used in our project.

B. Data Construction Details

Unlike contemporary state-of-the-art VLMs (e.g., Qwen3-VL [10], InternVL 3.5 [11]), our foundation dVLM, LLaDA-V, has not been extensively pre-trained on massive multimodal corpora. As a result, its world knowledge—particularly for driving—lags behind mainstream ARM-based VLMs. To align LLaDA-V with driving-specific knowledge, we curate and process data from existing driving datasets, focusing on supervision signals that emphasize scene understanding, high-level prediction and planning. Concretely, we aggregate examples from *ImpromptuVLA* [2], *DriveAction* [3], *DriveLM* [9], *DriveLMMo1* [4], and *CoVLA* [1] under the following rules: (i) for ImpromptuVLA, DriveLM, and DriveLMMo1, we retain only perception- and prediction-oriented instances, removing low-level planning tasks and any structured QA pairs that rely on special tokens; in addition, we keep only front-view examples to standardize visual context; (ii) for DriveAction, which provides high-quality QA pairs on human-like driving decisions, we convert multiple-choice answers into open-ended responses to promote more natural alignment; (iii) for CoVLA, which contains many Japanese driving videos ($\approx$ 40s each) with captions at 10 Hz, we mitigate redundancy by sampling only 2–3 clips per video and summarizing all captions within a clip into a single description using an LLM [10]. Following this pipeline, we obtain approximately **145k** driving-related QA pairs for alignment, providing targeted supervision that strengthens LLaDA-V's perception and prediction capabilities.

High–quality reasoning signals are essential for linking scene semantics to the ego vehicle's future motion [6, 13]. We therefore build a structured annotation pipeline tailored to trajectory prediction. Specifically, we construct reasoning annotations tailored to trajectory prediction by employing **GPT-4.1** [7] as an automatic annotator conditioned on the same signals available to the predictor—multi-frame front-view images, past ego state (e.g., velocity/acceleration and short waypoint history), a navigation command, and the ego's future waypoints—and by overlaying 2D bounding boxes on the current frame to strengthen object grounding. Inspired by Poutine [8], we structure the output into four fields where `object detection` lists salient agents and traffic elements, `explanation` provides a concise causal rationale linking those cues to the upcoming motion, `future behavior` specifies high-level longitudinal and lateral intentions consistent with the scene, and `trajectory` encodes the target future path as waypoints. At inference time the dVLM treats these four fields as templated slots and fills the corresponding values, yielding controllable and semantically grounded reasoning that remains consistent with the trajectory head. Finally, we annotate **23k** and **30k** reasoning data for nuScenes and WOD-E2E datasets, respectively.

C. Consistency Metric

Object↔Explanation Consistency. This metric measures how well the textual explanation matches the detected critical objects, and whether the explanation avoids hallucinating objects that are not present. For each example, we provide an LLM judge with a binary dictionary `critical_objects` (yes/no for each object type) and the model's explanation, together with the instruction prompt in Table 1. The judge first identifies which object types are clearly mentioned or implied, and which types are explicitly negated in the explanation, then compares them with the detected "yes" and "no" labels. We report two directional scores on a $[0, 5]$ scale: **O→E** (score_o2e) measures how many detected critical objects are actually mentioned in the explanation, and **E→O** (score_e2o) measures how many mentioned or negated objects are consistent with the detection labels. Both scores are normalized and rounded to one decimal, with higher values indicating better grounding and fewer hallucinations.

Behavior↔Trajectory Consistency. Given a future trajectory $\mathbf{T} = \{(x_t, y_t)\}_{t=1}^{N}$ in ego coordinates ($+x$ forward, $+y$ left), we first convert it into discrete lateral and longitudinal labels, then compare them with the model's predicted `meta_behaviour`. For the lateral behavior, we use the total path length L_{path} and the lateral displacement ratio to-

gether with the heading change:

$$r_{\text{lat}} = \frac{|y_N - y_1|}{L_{\text{path}} + \varepsilon},$$

$$\Delta\theta = \text{wrap}(\theta_{\text{end}} - \theta_{\text{start}}),$$

$$\theta_{\text{start}} = \text{atan2}(y_2 - y_1, x_2 - x_1),$$

$$\theta_{\text{end}} = \text{atan2}(y_N - y_{N-1}, x_N - x_{N-1}).$$

If r_{lat} is above a lane-change threshold and $|\Delta\theta|$ is small, we mark a left/right lane change (by the sign of $y_N - y_1$); if $|\Delta\theta|$ is larger than a turn threshold, we mark a left/right turn; otherwise we label the trajectory as `straight`.

For the longitudinal behavior, we compute average start and end speeds from the step-wise displacements $v_t = \|(x_t, y_t) - (x_{t-1}, y_{t-1})\|$:

$$v_{\text{start}} = \tfrac{1}{k} \sum_{t=2}^{k+1} v_t, \qquad v_{\text{end}} = \tfrac{1}{k} \sum_{t=N-k+1}^{N} v_t.$$

Small overall motion and near-zero v_{end} give `stop`; a clear increase from v_{start} to v_{end} gives `accelerate`; a clear decrease gives `decelerate`; otherwise we assign `keep`. We then check whether the model's lateral and longitudinal labels match these trajectory-derived labels (treating {`straight`, `lane_follow`, `keep_lane`} as equivalent laterally), yielding a binary indicator $m_{\text{cons}} \in \{0, 1\}$ for each sample. For WOD-E2E trajectories with N=20 future waypoints at 4 Hz (5 s horizon), we set k=4, so that v_{start} and v_{end} are computed over the first and last 1 s, respectively.

For each sample i in the evaluation set (M total samples), we compute binary indicators for longitudinal and lateral agreement:

$$m_{\text{lon}}^{(i)},\ m_{\text{lat}}^{(i)} \in \{0, 1\}.$$

The unweighted consistency scores are

$$\text{Con}_{\text{long}} = \frac{1}{M} \sum_{i=1}^{M} m_{\text{lon}}^{(i)}, \qquad \text{Con}_{\text{lat}} = \frac{1}{M} \sum_{i=1}^{M} m_{\text{lat}}^{(i)}.$$

To discount samples with inaccurate trajectories, we compute an ADE-based confidence weight for each sample i:

$$x^{(i)} = \max\big(0,\ \text{ADE}^{(i)} - 3.5\big), \qquad w^{(i)} = \frac{1}{1 + x^{(i)}/8.0}.$$

The weighted consistency scores (following our implementation) are

$$\text{Con}_{\text{long}} = \frac{1}{M} \sum_{i=1}^{M} w^{(i)} m_{\text{lon}}^{(i)}, \qquad \text{Con}_{\text{lat}} = \frac{1}{M} \sum_{i=1}^{M} w^{(i)} m_{\text{lat}}^{(i)}.$$

ADE weighting prevents the metric from being inflated by trivial predictions such as always outputting `keep` speed or `straight`/`lane_follow`, ensuring that high consistency reflects both correct behavior reasoning and accurate trajectory prediction.

Inference Setting. We perform inference using Fast-dLLM [12] with settings steps = 128, gen_length = 512, block_length = 512 and prefix_refresh_interval = 32, where steps specifies the number of generation iterations, gen_length sets the maximum number of tokens produced, block_length controls the size of each decoding block processed in parallel, and prefix_refresh_interval determines how frequently the prefix context cache is refreshed to balance speed and coherence.

D. Details of Training Paradigm

Stage I: We initialize the model with **LLaDA-V** and perform domain alignment on the collected driving corpus that mixes image-only and video samples. All images are resized to 384×384; for videos, we sample the past three frames at 2 Hz to form a short temporal context. Training follows the objective in Eq. (2). We use AdamW with a learning rate of 1×10^{-5}, batch size 64, and train for 1 epoch.

Stage II: Starting from the Stage I checkpoint, we finetune on **nuScenes** ($\mathcal{V}_{\text{nuScenes}}$) and **WOD-E2E** ($\mathcal{V}_{\text{waymo}}$), using the same input preprocessing (images at 384×384; videos sampled at 2 Hz with three past frames). The structured training objective in Eq. (3) couples structured reasoning supervision with planning targets. We adopt AdamW (learning rate 1×10^{-5}, batch size 64) and train for 3 epochs. All experiments are run on 8× NVIDIA A100 80 GB GPUs; Stage I takes about 14 hours and Stage II about 6 hours.

ARM-based VLM: To fairly compare with ARM-based VLMs under matched conditions, we also start from **LLaVA-OneVision** [5] as a foundation model with comparable general performance and pretraining scale, and train it using the same two-stage recipe and hyperparameters to obtain **VLM-AD** for comparison.

E. Demonstration

In this section, we provide qualitative case studies illustrating how *d*VLM-AD improves reasoning–action consistency over VLM-AD across diverse scenarios, including cluttered intersections in dense fog (Fig. 1), multi-vehicle urban junctions with stop signs (Fig. 2), open-road speed-hump navigation (Fig. 3), construction-constrained downtown intersections (Fig. 4), nighttime left-turn commands (Fig. 5), and encounters with oncoming traffic (Fig. 6).

F. Prompts

We summarize the core prompts used in our pipeline in this section. Table 1 provides the LLM-judge instruction for Object↔Explanation Consistency, while Tables 2 and 3 show the annotation prompts used to generate reasoning labels on nuScenes and WOD-E2E Dataset, respectively.

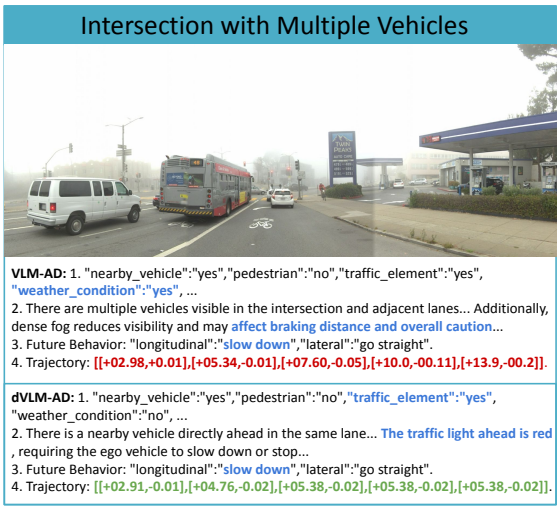

Figure 1. Intersection with multiple vehicles in dense fog, where dVLM-AD shows more consistent "slow down" behavior than the VLM-AD.

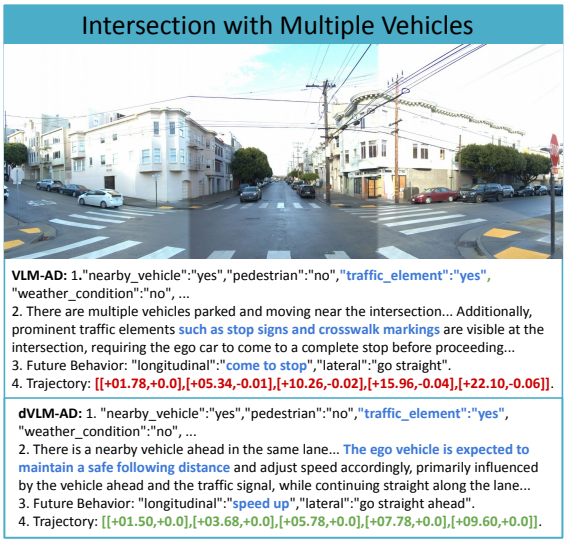

Figure 2. Urban intersection with multiple vehicles and stop signs, where dVLM-AD more faithfully couples its reasoning about stopping and following distance with the generated trajectory.

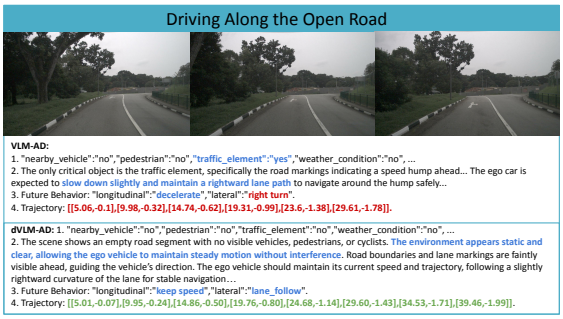

Figure 3. Driving along an open road with a speed-hump marking, where dVLM-AD's lane-following trajectory better matches its reasoning about the clear road than the VLM-AD.

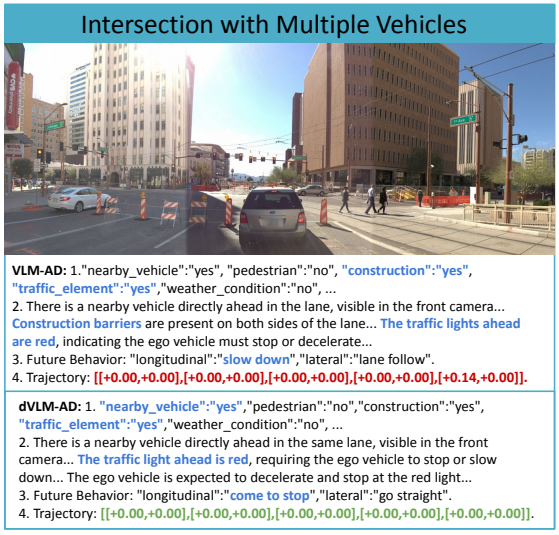

Figure 4. Downtown intersection with construction barriers and a red light, where dVLM-AD's stopping behavior at the signal is more consistent with its textual reasoning than that of VLM-AD.

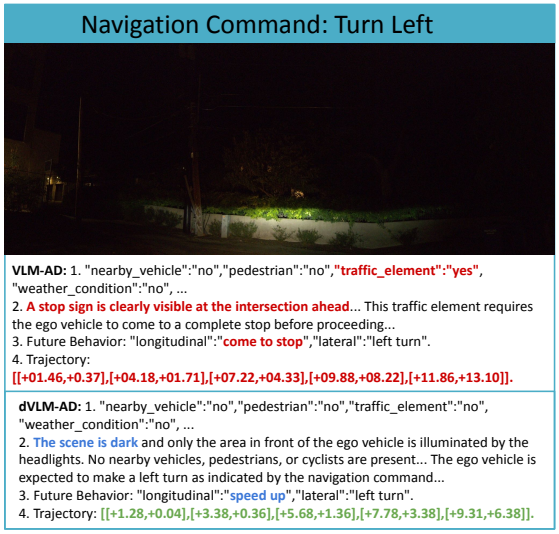

Figure 5. Nighttime scenario with a left-turn navigation command, where dVLM-AD executes a left-turn trajectory aligned with its reasoning while the VLM-AD behaves more conservatively.

Figure 6. Scenario with oncoming traffic, where dVLM-AD's reasoning process is more accurate with less hallucination.

LLM Judge Instruction for Object↔Explanation Consistency

```
You are an evaluator. Judge two-way consistency between:

critical_objects = {CRIT}
explanation = {EXPL}

------------------
Object Definitions
------------------
Detect mentions using synonyms/paraphrases and clear implications:
- nearby_vehicle: car, vehicle, truck, bus, lead/parked vehicle, traffic ahead
- pedestrian: person, walker, human crossing
- cyclist: bicyclist, bike rider
- construction: roadwork, cones, barriers, work zone
- traffic_element: traffic light/signal, stop/yield/speed-limit/dir/lane sign
- weather_condition: rain, snow, fog, wet/slippery road
- road_hazard: debris, obstacle, pothole, spill
- emergency_vehicle: ambulance, fire truck, police car, siren vehicle
- animal: dog, deer, animal on/near road
- special_vehicle: bus, trailer truck, large commercial vehicle
- conflicting_vehicle: oncoming/crossing/merging vehicle conflicting with ego path
- door_opening_vehicle: parked car door opening into lane

-----------------------------
Scoring (0-5, one decimal)
-----------------------------
Interpretation of "mentioned or implied":
- Synonyms, paraphrases, and clear hyponyms/hypernyms count.
- Strong indirect cues count (e.g., "stopped at a stop sign" → traffic_element).
- Explicit negation (e.g., "no pedestrians") maps to label "no".
- Coreference continues a mention.
- Ambiguous mentions do NOT count; if unsure, treat as NOT mentioned.

Sets:
- From explanation:
  M+ = {objects clearly PRESENT/IMPLIED}
  M- = {objects explicitly ABSENT via negation}
- From critical_objects:
  Y  = {k | critical_objects[k] == "yes"}

Multi-mapping: if a phrase reasonably maps to multiple categories
(e.g., "bus"), count as present if at least one mapped category is "yes".

Directional scores:
- score_o2e (Object → Explanation):
    N_yes = |Y|, C_yes = |Y  M+|
    If N_yes == 0 → score_o2e = 5.0
    Else score_o2e = round(5 * C_yes / N_yes, 1)

- score_e2o (Explanation → Object):
    P = |M+|,  A = # in M+ correctly labeled "yes"
    N_neg = |M-|, B = # in M- correctly labeled "no"
    If (P + N_neg) == 0 → score_e2o = 5.0
    Else score_e2o = round(5 * (A + B) / (P + N_neg), 1)

Strict rule:
- If unsure about a mapping or implication, do NOT credit it.
- Clamp both scores to [0.0, 5.0], one decimal.

------------------
Output (JSON only)
------------------
{
  "score_o2e": 0.0,
  "score_e2o": 0.0,
  "missing_in_explanation": [],
  "missing_in_critical_objects": [],
  "notes": "short reason"
}
```

Table 1. Prompt for the LLM judge used to compute Object↔Explanation Consistency (Section C).

Annotation Assistant Prompt for nuScenes Dataset

```
You are an autonomous-driving annotation assistant.

You will receive:
(1) Three front-view frames at timestamps [-1.0s, -0.5s, 0.0s], where 0.0s is the current frame.
(2) 2D bounding boxes ON THE CURRENT FRAME ONLY.
(3) A high-level navigation command (may be synonyms like "go straight").
(4) A 3-second future trajectory (teacher) with 6 waypoints in meters. The future trajectory is represented in the ego-vehicle coordinate system,
where the +x axis points forward (the direction the car is facing), and the +y axis points to the left side of the vehicle.

Your task:
Fill the STRICT JSON below with the correct values based on the inputs.
FOCUS ONLY on:
- critical_objects (binary yes/no)
- explanation (concise reason, no step-by-step)
- meta_behaviour (speed, command)
Do NOT include any "trajectory" field in the output.

Decision rules (apply consistently):
- Use ONLY the CURRENT FRAME's bounding boxes to judge objects.
- critical_objects.* must be "yes" only if it may influence ego behavior within the next 3 seconds.
  Definitions:
  * nearby_vehicle: vehicles within ~0-30 m or directly affecting lane choice/gap keeping.
  * conflicting_vehicle: objects whose paths or right-of-way conflict with ego's trajectory-defined path.
  * traffic_element: signals, signs, markings, stop lines, etc., that constrain behavior now.
  * pedestrian/cyclist: may cross or enter ego path imminently.
  * construction/weather_condition/road_hazard/emergency_vehicle/animal/special_vehicle/door_opening_vehicle: mark "yes" only if relevant
  in the next 3 s.
- Speed label from the future trajectory's forward progress:
  * increasing forward step  "accelerate"
  * decreasing  "decelerate"
  * roughly constant  "keep"
  * speed equals zero constant  "stop"
  * unclear  "other"
- Use ONLY the CURRENT FRAME's bounding boxes to judge objects.
- For meta_behaviour.command, primarily INFER FROM THE FUTURE TRAJECTORY SHAPE:
  * small lateral displacement & near-zero curvature & the current lane itself is curved (e.g., highway ramp, continuous bend)  "straight"
  or "lane_follow"
  * left-turn arc (positive leftward displacement/curvature)  "left_turn" or "left_lane_change" or "overtake"
  * right-turn arc (negative leftward displacement/curvature)  "right_turn" or "right_lane_change" or "overtake"
  If the provided navigation command conflicts with the trajectory, FOLLOW THE TRAJECTORY.

STRICT language & privacy constraints (no exceptions):
- Output STRICT JSON only; no extra keys, no comments, no markdown.
- "critical_objects.*" values must be exactly "yes" or "no".
- Write a concise paragraph (~100 words) describing (DO NOT mention or allude to any provided ground-truth/meta data: FORBIDDEN terms/concepts
include "trajectory", "future trajectory", "planned path/arc", "teacher", "ground truth", "inputs provided", "bounding boxes", "frames/timestamps",
"metadata", "leftward progression". Use phrasing like "the ego car is expected", "ego car should take", etc.):
    1. The location and attributes of the identified critical objects or conditions (only mention classes marked "yes").
    2. How the future behaviors of these objects/conditions affect the ego vehicle's next 3-second behavior. Focus on objects' future behavior,
    interactions with the ego car, traffic rules, and ego dynamics. Mention only the classes marked "yes".
- "meta_behaviour.speed"  {keep, accelerate, decelerate, stop, other}
- "meta_behaviour.command"  {straight, yield, lane_follow, left_turn, right_turn, right_lane_change, left_lane_change, reverse, overtake, other}
- Do NOT output any "trajectory" key.

RETURN FORMAT (keep structure identical; fill values only):
{
  "critical_objects": {
    "nearby_vehicle": ,
    "pedestrian": ,
    "cyclist": ,
    "construction": ,
    "traffic_element": ,
    "weather_condition": ,
    "road_hazard": ,
    "emergency_vehicle": ,
    "animal": ,
    "special_vehicle": ,
    "conflicting_vehicle": ,
    "door_opening_vehicle":
  },
  "explanation": "...",
  "meta_behaviour": {
    "speed": "",
    "command": ""
  }
}

INPUTS:
- 3 Frames: -1.0s, -0.5s, 0.0s (current)
- Navigation command (string): {NAV_COMMAND}
- Future trajectory (teacher) [6×[x,y] meters; +x forward, +y leftward; A positive y displacement means the vehicle is moving leftward --
typically indicating a left turn or lane change to the left.
A negative y displacement means the vehicle is moving rightward -- typically indicating a right turn or lane change to the right]:
{FUTURE_TRAJ_ARRAY}

OUTPUTS (json format):
```

Table 2. Prompt used with Claude-Sonnet-3.7 to generate reasoning annotations on the nuScenes dataset.

Annotation Assistant Prompt for WOD-E2E Dataset

```
You are an autonomous-driving annotation assistant.

You will receive:
(1) Three synchronized current-frame images from the ego vehicle: left_front, front, right_front (visual context to detect
relevant objects/conditions).
(2) A high-level navigation command (may include synonyms, e.g., “go straight”).
(3) A 3.0-second history ego state with 7 samples at 0.5s intervals from t-3.0s to t-0.0s. Each sample provides position [x, y] in meters
(+x forward, +y leftward), velocity [vx, vy] in m/s, and acceleration [ax, ay] in m/s².
(4) A 5.0-second future trajectory (teacher) with 10 waypoints in meters, sampled every 0.5s from t+0.0s to t+4.5s (+x forward, +y leftward).

Your task:
Fill the STRICT JSON below with the correct values based on the inputs.
FOCUS ONLY on:
- critical_objects (binary yes/no)
- explanation (concise reason, no step-by-step)
- meta_behaviour (longitudinal, lateral)
Do NOT include any "trajectory" field in the output.

Decision rules (apply consistently):
- Use ONLY the CURRENT FRAMES to judge objects.
- critical_objects.* must be "yes" only if it may influence ego behavior within the next 5 seconds.
  Definitions:
  * nearby_vehicle: vehicles within ~0-30 m or directly affecting lane choice/gap keeping.
  * conflicting_vehicle: objects whose paths or right-of-way conflict with ego’s trajectory-defined path.
  * traffic_element: signals, signs, markings, stop lines, etc., that constrain behavior now.
  * pedestrian/cyclist: may cross or enter ego path imminently.
  * construction/weather_condition/road_hazard/emergency_vehicle/animal/special_vehicle/door_opening_vehicle: mark "yes" only if relevant
  in the next 5 s.
- Speed label from the future trajectory’s forward progress:
  * increasing forward step  "accelerate"
  * decreasing  "decelerate"
  * roughly constant  "keep"
  * speed equals zero constant  "stop"
  * unclear  "other"
- Use ONLY the CURRENT FRAMES to judge objects.
- For meta_behaviour.command, primarily INFER FROM THE FUTURE TRAJECTORY SHAPE:
  * small lateral displacement & near-zero curvature & the current lane itself is curved (e.g., highway ramp, continuous bend)  "straight"
  or "lane_follow"
  * left-turn arc (positive leftward displacement/curvature)  "left_turn" or "left_lane_change" or "overtake"
  * right-turn arc (negative leftward displacement/curvature)  "right_turn" or "right_lane_change" or "overtake"
  If the provided navigation command conflicts with the trajectory, FOLLOW THE TRAJECTORY.

STRICT language & privacy constraints (no exceptions):
- Output STRICT JSON only; no extra keys, no comments, no markdown.
- "critical_objects.*" values must be exactly "yes" or "no".
- "explanation" Write a concise paragraph (~100 words) that only discusses the objects or conditions marked “yes”. The paragraph must include: (1)
a brief description of the object(s) or condition(s), including location and relevant visual cues such as brake lights, turn signals,
pedestrian posture, lane obstruction, etc.; (2) a clear statement of the expected longitudinal and lateral motion of these objects or conditions,
such as slowing down, cutting across, merging, or blocking lane space; (3) an explanation of how these expected behaviors will influence the
ego vehicle over the next 5 seconds, including possible actions such as slowing, yielding, lane keeping, changing lanes, or stopping.
Do not reference or imply any hidden data sources such as “trajectory”, “future path”, “ground truth”, “teacher”, “metadata”, or “inputs provided”.
Use natural phrasing such as “the ego car is expected to...” rather than referencing annotations or datasets.
- "meta_behaviour.longitudinal"  {keep, accelerate, decelerate, stop, other}
- "meta_behaviour.lateral"  {straight, yield, lane_follow, left_turn, right_turn, right_lane_change,
left_lane_change, reverse, overtake, other}
- Your meta behavior should not rely on navigation commands. Instead, it should primarily reference trajectory and visual information. For example,
if the navigation command is “go straight,” but there is a vehicle ahead that you should overtake, then the meta behavior should be
lane_change_left or right_change_left.

RETURN FORMAT (keep structure identical; fill values only):
{
  "critical_objects": {
    "nearby_vehicle": ,
    "pedestrian": ,
    "cyclist": ,
    "construction": ,
    "traffic_element": ,
    "weather_condition": ,
    "road_hazard": ,
    "emergency_vehicle": ,
    "animal": ,
    "special_vehicle": ,
    "conflicting_vehicle": ,
    "door_opening_vehicle":
  },
  "explanation": "...",
  "meta_behaviour": {
    "longitudinal": "",
    "lateral": ""
  }
}

INPUTS:
- current-frame images from the ego vehicle: left_front, front, right_front cameras
- Navigation command (string): {NAV_COMMAND}
- History ego state: {EGO_STATE}
- Future trajectory (teacher) [6×[x,y] meters; +x forward, +y leftward]:
{FUTURE_TRAJ_ARRAY}

Notes:
- Consider only influences within the next 5 seconds when deciding critical_objects and behavior.
- Coordinates and velocities use +x forward, +y leftward convention.
- The three camera views provide visual context for object detection and may change the intended path (e.g., obstacle ahead ->
yield/stop/avoidance or lane change).

OUTPUTs (json format):
```

Table 3. Prompt used with Claude-Sonnet-3.7 to generate reasoning annotations on the WOD-E2E dataset.

References

[1] Hidehisa Arai, Keita Miwa, Kento Sasaki, Kohei Watanabe, Yu Yamaguchi, Shunsuke Aoki, and Issei Yamamoto. Covla: Comprehensive vision-language-action dataset for autonomous driving. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1933–1943. IEEE, 2025. 1

[2] Haohan Chi, Huan-ang Gao, Ziming Liu, Jianing Liu, Chenyu Liu, Jinwei Li, Kaisen Yang, Yangcheng Yu, Zeda Wang, Wenyi Li, et al. Impromptu vla: Open weights and open data for driving vision-language-action models. *arXiv preprint arXiv:2505.23757*, 2025. 1

[3] Yuhan Hao, Zhengning Li, Lei Sun, Weilong Wang, Naixin Yi, Sheng Song, Caihong Qin, Mofan Zhou, Yifei Zhan, and Xianpeng Lang. Driveaction: A benchmark for exploring human-like driving decisions in vla models. *arXiv preprint arXiv:2506.05667*, 2025. 1

[4] Ayesha Ishaq, Jean Lahoud, Ketan More, Omkar Thawakar, Ritesh Thawkar, Dinura Dissanayake, Noor Ahsan, Yuhao Li, Fahad Shahbaz Khan, Hisham Cholakkal, Ivan Laptev, Rao Muhammad Anwer, and Salman Khan. Drivelmm-o1: A step-by-step reasoning dataset and large multimodal model for driving scenario understanding, 2025. 1

[5] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024. 2

[6] NVIDIA, :, Yan Wang, Wenjie Luo, Junjie Bai, Yulong Cao, Tong Che, Ke Chen, Yuxiao Chen, Jenna Diamond, Yifan Ding, Wenhao Ding, Liang Feng, Greg Heinrich, Jack Huang, Peter Karkus, Boyi Li, Pinyi Li, Tsung-Yi Lin, Dongran Liu, Ming-Yu Liu, Langechuan Liu, Zhijian Liu, Jason Lu, Yunxiang Mao, Pavlo Molchanov, Lindsey Pavao, Zhenghao Peng, Mike Ranzinger, Ed Schmerling, Shida Shen, Yunfei Shi, Sarah Tariq, Ran Tian, Tilman Wekel, Xinshuo Weng, Tianjun Xiao, Eric Yang, Xiaodong Yang, Yurong You, Xiaohui Zeng, Wenyuan Zhang, Boris Ivanovic, and Marco Pavone. Alpamayo-r1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail, 2025. 1

[7] OpenAI. Introducing gpt-4.1 in the api, 2025. OpenAI. 1

[8] Luke Rowe, Rodrigue de Schaetzen, Roger Girgis, Christopher Pal, and Liam Paull. Poutine: Vision-language-trajectory pre-training and reinforcement learning post-training enable robust end-to-end autonomous driving. *arXiv preprint arXiv:2506.11234*, 2025. 1

[9] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering. In *European conference on computer vision*, pages 256–274. Springer, 2024. 1

[10] Qwen Team. Qwen3 technical report, 2025. 1

[11] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025. 1

[12] Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. *arXiv preprint arXiv:2505.22618*, 2025. 2

[13] Zewei Zhou, Tianhui Cai, Seth Z Zhao, Yun Zhang, Zhiyu Huang, Bolei Zhou, and Jiaqi Ma. Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning. *arXiv preprint arXiv:2506.13757*, 2025. 1