VGGDrive: Empowering Vision-Language Models with Cross-View Geometric Grounding for Autonomous Driving

Supplementary Material

To further supplement the main content of this paper, we have organized additional material to elaborate on some key details. Specifically, Sec. A provides further clarification on the application of VGGDrive across five mainstream autonomous driving benchmarks and the corresponding evaluation metrics. Sec. B presents an interesting ablation analysis on the NAVSIM closed-loop trajectory planning task, where 3D features V^{3d} are injected into different decoding layers $\{DL_i\}_{i=1}^{n}$ of the VLM. Sec. C presents additional visual samples that highlight the model's performance on critical tasks in autonomous driving. Tab. S1 presents the training and testing sample statistics of five mainstream autonomous driving datasets used by VGGDrive, along with the model capabilities evaluated on each dataset.

A. Dataset and Metric

To assess the performance of VGGDrive across various attributes within the autonomous driving domain, we conduct evaluations on five prominent benchmarks. These benchmarks cover task-specific scenarios, including scene understanding, cross-view risk object perception, action and state prediction, and trajectory planning. In the tables within the main text, we highlight key metrics related to the model's cross-view 3D capabilities by using bold formatting.

NAVSIM. This dataset [5] is a real-world, planning-focused dataset derived from OpenScene, which is a compact version of nuPlan, the largest publicly available annotated driving dataset (1,192 and 136 scenarios for training and testing). NAVSIM is designed to evaluate the performance of autonomous driving systems in complex, dynamic scenarios. It utilizes a combination of eight cameras providing a 360° field of view (FOV) and a merged LiDAR point cloud from five sensors. NAVSIM specifically targets challenging driving situations with dynamic driving intentions, while deliberately excluding simpler, static scenarios such as stationary scenes or constant-speed driving. The NAVSIM benchmark provides a nonreactive simulation environment and employs the Predictive Driver Model Score (PDMS) as its closed-loop planning metric:

$$PDMS = NC \times DAC \times \left(\frac{5\times EP+5\times TTC+2\times C}{12}\right) \tag{11}$$

where PDMS integrates five sub-metrics: No At-Fault Collision (NC), Drivable Area Compliance (DAC), Time-to Collision (TTC), Comfort (C), and Ego Progress (EP) to produce a comprehensive closed-loop planning score.

NuInstruct. This dataset [6] samples a total of 11,850 keyframes from 850 videos within the NuScenes dataset. It includes a wide range of challenging samples, such as cross-view risk perception, target distance estimation, agent and ego state prediction, target motion prediction, and reasoning tasks. The dataset effectively reflects the model's ability to perform cross-view understanding and analyze 3D geometric scene information. NuInstruct uses a variety of metrics to evaluate different tasks, including Mean Absolute Error (MAE) for regression tasks, accuracy for classification tasks, Mean Average Precision (mAP) for detection tasks, and a rule-based BLEU metric for captioning tasks.

DriveLM. This dataset [39] is built upon the real-world driving dataset nuScenes and covers interactive scenarios involving vehicles, pedestrians, and traffic infrastructure on urban roads. The keyframes focus on moments that mark changes in driving intentions, such as acceleration, deceleration, and turning. The data samples not only include "what objects are currently present" (cross-view perception), but also encompass "how objects will move in the future" (action & states prediction), "what the vehicle should do" (planning), "specific behavior classifications" (e.g., fast driving straight, slow right turn), and "trajectory coordinates" (motion). This dataset similarly reflects the model's ability to perform cross-view perception and analyze 3D geometric scene understanding. DriveLM implements four evaluation metrics: accuracy, LLM score, language rule-

Table S1. Training and Testing Sample Statistics of Five Mainstream Autonomous Driving Datasets Used by VGGDrive, and the Model Capabilities Evaluated on Each Dataset

Dataset	Train Samples	Test Samples	Cross-view perception	Action&States prediction	Trajectory planning	Scene understanding
NAVSIM [5]	103.3k	12.1k			✓	
NuInstruct [6]	71.8k	16.1k	✓	✓		
DriveLM [39]	376.2k	15.5k	✓	✓		✓
OmniDrive [44]	318.4k	54.1k				✓
nuScenes [30]	28.1k	6.0k			✓	

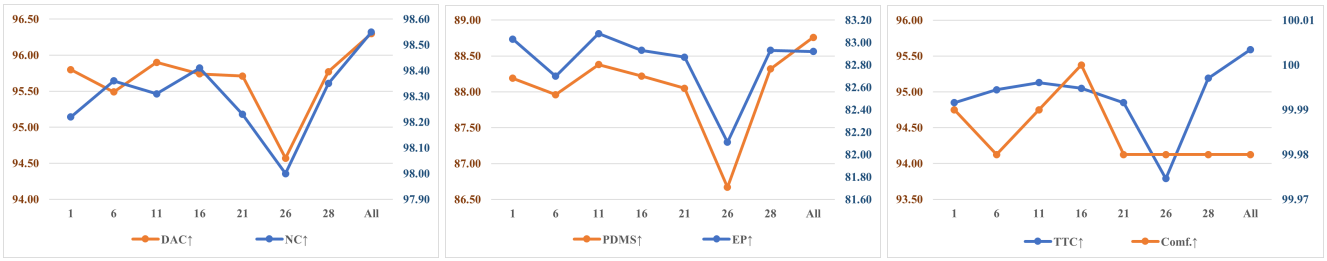


Figure S1. Ablation analysis of closed-loop trajectory planning performance on the NAVSIM dataset when cross-view 3D geometric empowerment and adaptive injection are applied to individual decoding layers of the LLM.

based evaluation, and match score.

OmniDrive. This dataset [44] is also built upon the nuScenes dataset and primarily focuses on scene description, attention-based question answering, counterfactual reasoning question answering, decision planning question answering, and general dialogue. Due to the limitations in the availability of evaluation data and metrics, we selected scene description and general dialogue from this dataset to train and evaluate the model's ability to understand autonomous driving scene captions. The dataset primarily includes three language evaluation metrics: BLEU, CIDEr, and ROUGE, with the "average" representing the mean of the three metrics.

nuScenes. We conduct open-loop trajectory planning experiments on the challenging public nuScenes dataset [30], which contains 1,000 driving scenes, each lasting approximately 20 seconds. The scene images are captured by six cameras, providing a 360° horizontal field of view (FOV), with keyframes annotated at a frequency of 2Hz. We follow the standard training and testing setup, adhering to the evaluation metrics used in OmniDrive. Among these metrics [30, 44], L2 and Collision Rate are widely used, and we additionally introduce the Intersection Rate, which calculates the rate of collisions or intersections between the predicted trajectories and curbs (road boundaries).

B. VLM Layer Injection Ablation

To validate the effectiveness of our hierarchical adaptive injection mechanism and explore the performance of individual decoding layers when injected with 3D features, we conduct further ablation experiments. Specifically, while maintaining identical configurations as the final VGGDrive model, we perform cross-view 3D geometric feature empowerment and adaptive injection on individual decoding layers $\{DL_i\}_{i=1}^{n}$. For the Qwen2.5-VL-7B model, which contains 28 decoding layers (i.e., $n = 28$), we select different layers at regular intervals and observe the performance of closed-loop trajectory planning on the NAVSIM dataset.

1) Comparison of Fig. S1 and Tab. 1 (in the main text): Using our cross-view 3D Geometric Enabler and decoupled adaptive injection mechanism, a significant performance improvement is observed when injecting features into a single layer (PDMS around 88), compared to the base VLM model (PDMS = 86.04). Meanwhile, this method also outperforms existing VGGT and VLM integration solutions (PDMS < 87). Compared to single-layer injection, our VGGDrive achieves even better performance through full-layer adaptive injection. These results highlight the positive impact of VGGT features for multi-view tasks in autonomous driving and validate the effectiveness and robustness of our proposed VGGDrive design.

2) Analysis of performance across different layers: By analyzing the performance variations across different layers, we observe significant differences in closed-loop planning performance when injecting 3D features into different decoding layers. Overall, a peak performance is achieved around layer 11, with reasonable performance maintained at both ends. The observed variations across layers also offer valuable insights for future research, suggesting that a reasonable trade-off between efficiency and performance could be achieved by leveraging just a single layer.

C. Qualitative Results

This section further presents several visual examples to demonstrate VGGDrive's capability in trajectory planning and action or state prediction within complex autonomous driving scenarios. Fig. S2, S3, and S4 show trajectory planning visualizations for left turns, right turns, and straight driving, respectively, in the NAVSIM benchmark. Fig. S5 provides additional examples of VGGDrive's open-loop trajectory performance in the nuScenes benchmark. In Fig. S6, we further illustrate how VGGDrive predicts the states and actions of both the ego vehicle and surrounding agents after perceiving and modeling cross-view scenes. These visualizations effectively highlight VGGDrive's advantage in injecting VGGT's cross-view 3D scene features into VLM-driven autonomous driving tasks, showcasing substantial performance improvements. This validates the effectiveness and promising prospects of this novel technical approach, which avoids the reliance on constructing large-scale VQA datasets or adding additional trajectory-generation action decoders.

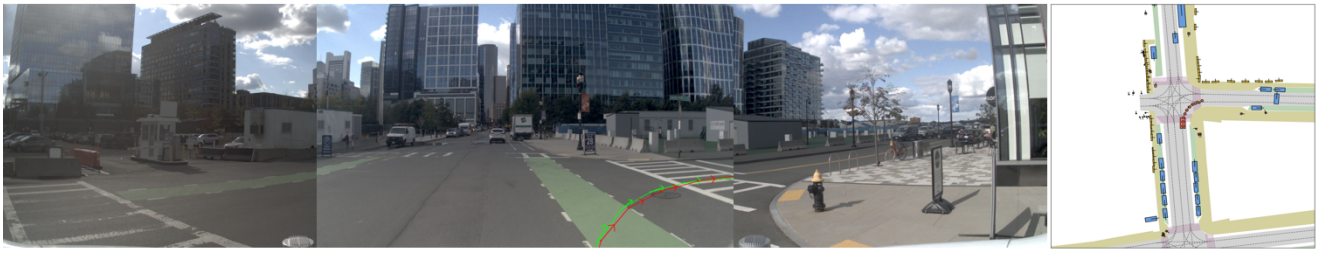
“As an autonomous driving system, predict the vehicle‘s trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-12.39, -0.71] - t-2: [-8.23, -0.51] - t-1: [-4.1, -0.29] - t-0: [0.0, 0.0]; 3. Steering: 0.44 degrees; 4. Speed: 8.24 m/s; 5. Acceleration: -0.02 m/s²; 6. Active navigation command: Front Left. Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"

"As an autonomous driving system, predict the vehicle's trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-10.56, 1.7] - t-2: [-7.14, 0.79] - t-1: [-3.61, 0.19] - t-0: [0.0, 0.0]; 3. Steering: 4.77 degrees; 4. Speed: 7.22 m/s; 5. Acceleration: 0.09 m/s²; 6. Active navigation command: Turn Left. Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"

"As an autonomous driving system, predict the vehicle's trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-8.5, 1.42] - t-2: [-5.68, 0.7] - t-1: [-2.83, 0.19] - t-0: [0.0, 0.0]; 3. Steering: 6.96 degrees; 4. Speed: 5.63 m/s; 5. Acceleration: -0.19 m/s²; 6. Active navigation command: Turn Left; Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"

“As an autonomous driving system, predict the vehicle‘s trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-12.88, 2.47] - t-2: [-8.77, 1.15] - t-1: [-4.45, 0.31] - t-0: [0.0, 0.0]; 3. Steering: 3.88 degrees; 4. Speed: 8.93 m/s; 5. Acceleration: 0.19 m/s²; 6. Active navigation command: Front Left. Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"

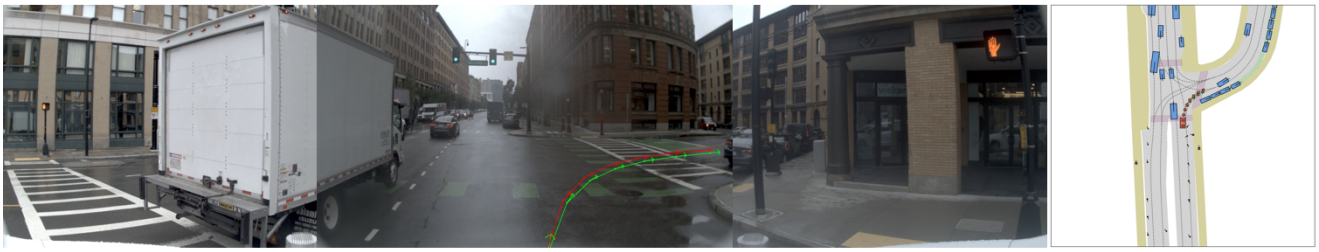
Figure S2. Qualitative results on the closed-loop trajectory planning task in the Navtest benchmark are presented, showcasing a typical left-turn example to demonstrate the performance of VGGDrive in complex driving scenarios.



"As an autonomous driving system, predict the vehicle's trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-7.48, -0.4] - t-2: [-4.75, -0.21] - t-1: [-2.27, -0.05] - t-0: [0.0, 0.0]; 3. Steering: -4.62 degrees; 4. Speed: 4.43 m/s; 5. Acceleration: -0.65 m/s²; 6. Active navigation command: Turn Right; Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"

"As an autonomous driving system, predict the vehicle's trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-3.73, -0.1] - t-2: [-2.28, -0.05] - t-1: [-1.1, -0.02] - t-0: [0.0, 0.0]; 3. Steering: -2.73 degrees; 4. Speed: 2.27 m/s; 5. Acceleration: -0.01 m/s²; 6. Active navigation command: Turn Right. Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"

"As an autonomous driving system, predict the vehicle's trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-6.12, -0.24] - t-2: [-4.09, -0.14] - t-1: [-2.03, -0.05] - t-0: [0.0, 0.0]; 3. Steering: -3.49 degrees; 4. Speed: 4.13 m/s; 5. Acceleration: 0.09 m/s²; 6. Active navigation command: Turn Right. Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"



"As an autonomous driving system, predict the vehicle's trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-8.88, -0.3] - t-2: [-5.84, -0.17] - t-1: [-2.87, -0.05] - t-0: [0.0, 0.0]; 3. Steering: -2.24 degrees; 4. Speed: 5.66 m/s; 5. Acceleration: -0.37 m/s²; 6. Active navigation command: Turn Right. Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"

Figure S3. Qualitative results on the closed-loop trajectory planning task in the Navtest benchmark are presented, showcasing a typical right-turn example to demonstrate the performance of VGGDrive in complex driving scenarios.

"As an autonomous driving system, predict the vehicle's trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-12.09, -0.0] - t-2: [-7.82, -0.01] - t-1: [-3.77, -0.0] - t-0: [0.0, 0.0]; 3. Steering: 0.08 degrees; 4. Speed: 7.18 m/s; 5. Acceleration: -1.28 m/s²; 6. Active navigation command: Go Straight. Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"

"As an autonomous driving system, predict the vehicle's trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-7.21, 0.13] - t-2: [-5.26, 0.06] - t-1: [-2.85, 0.0] - t-0: [0.0, 0.0]; 3. Steering: 0.49 degrees; 4. Speed: 6.02 m/s; 5. Acceleration: 1.52 m/s²; 6. Active navigation command: Go Straight. Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"

"As an autonomous driving system, predict the vehicle's trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-14.56, 0.02] - t-2: [-9.76, -0.01] - t-1: [-4.86, -0.01] - t-0: [0.0, 0.0]; 3. Steering: 0.14 degrees; 4. Speed: 9.66 m/s; 5. Acceleration: -0.20 m/s²; 6. Active navigation command: Go Straight. Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory- Maintain numerical precision to 2 decimal places"

"As an autonomous driving system, predict the vehicle's trajectory based on: 1. Visual perception from front-left, front, and front-right camera views; 2. Historical motion context (last 4 timesteps): - t-3: [-8.51, 0.03] - t-2: [-5.71, 0.03] - t-1: [-2.85, 0.01] - t-0: [0.0, 0.0]; 3. Steering: 0.32 degrees; 4. Speed: 5.66 m/s; 5. Acceleration: -0.10 m/s²; 6. Active navigation command: Go Straight. Output requirements: - Predict 8 future trajectory points - Each point format: [x:float, y:float] - Use [...] to encapsulate the trajectory - Maintain numerical precision to 2 decimal places"

Figure S4. Qualitative results on the closed-loop trajectory planning task in the Navtest benchmark are presented, showcasing a typical straight-ahead example to demonstrate the performance of VGGDrive in complex driving scenarios.

Prompt: "-Longitudinal velocity: -0.12 m/s - Lateral velocity: 3.52 m/s - Longitudinal acceleration: 0.15 m/s - Lateral acceleration: 2.32 m/s -Yaw rate: -0.00 rad/s - Vehicle length: 4.08 m - Vehicle width: 1.85 m - Speed magnitude: 4.59 m/s - Steering angle: -2.69 degrees - Navigation Information = Turn Left What trajectory comes next?"

GT: [[2.67, 0.25], [4.91, 0.9], [7.15, 1.93], [9.41, 3.29], [11.83, 4.99], [14.5, 6.97]]

Pre: [[2.3, 0.2], [4.67, 0.89], [7.08, 2.01], [9.55, 3.52], [12.1, 5.38], [14.79, 7.5]]

Prompt: "- Longitudinal velocity: 0.23 m/s - Lateral velocity: 7.30 m/s - Longitudinal acceleration: -1.34 m/s - Lateral acceleration: -0.61 m/s - Yaw rate: 0.02 rad/s - Vehicle length: 4.08 m - Vehicle width: 1.85 m - Speed magnitude: 7.14 m/s - Steering angle: 0.23 degrees - Navigation Information = Turn Right; What trajectory comes next?"

GT: [[3.39, -0.04], [6.39, -0.22], [8.59, -0.53], [11.23, -1.23], [13.88, -2.43], [16.37, -4.21]]

Pre: [[3.49, -0.07], [6.77, -0.29], [9.87, -0.7], [12.79, -1.37], [15.53, -2.34], [18.09, -3.6]]

Figure S5. Qualitative results on the open-loop trajectory planning task in the nuScenes benchmark are presented.

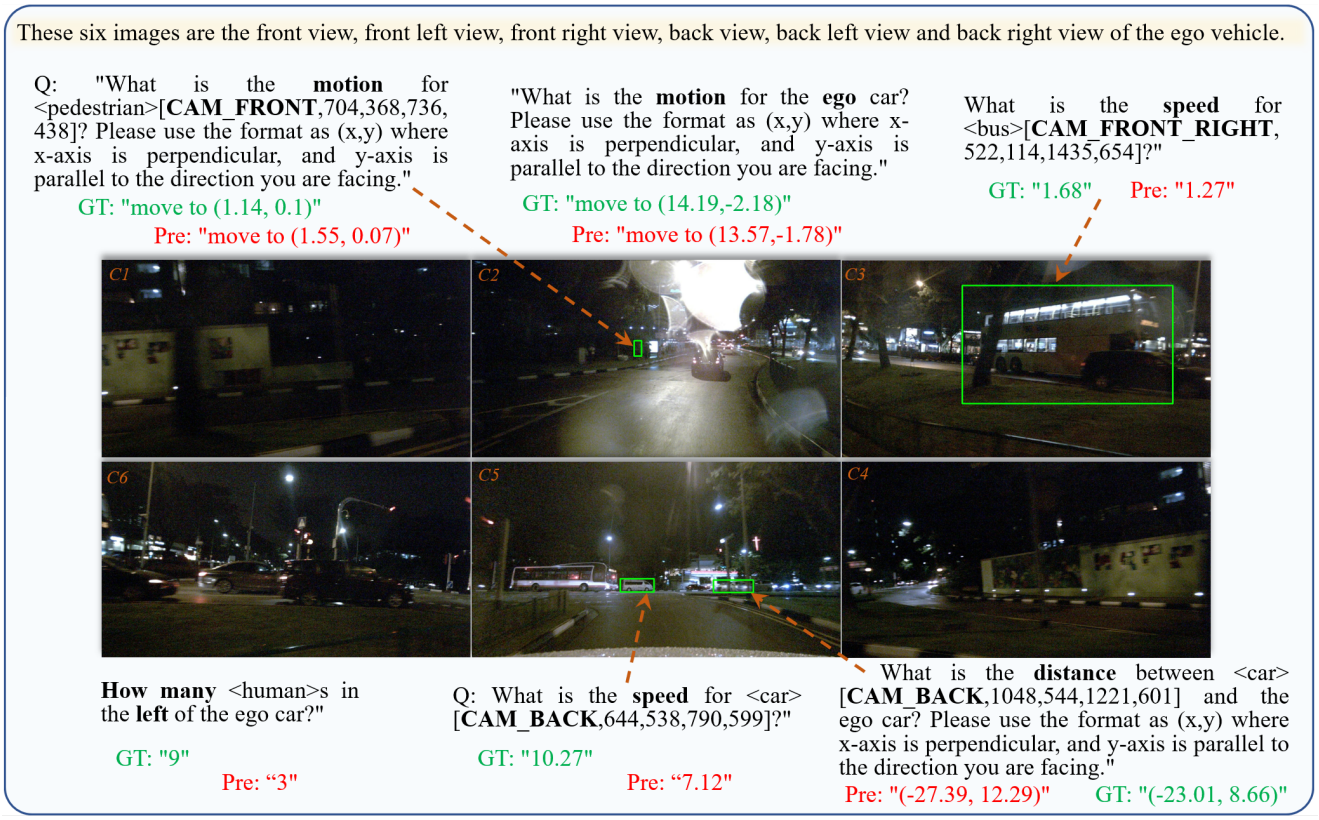


Figure S6. Qualitative results on Action & States prediction in autonomous driving scenarios are presented.