VLG-DRIVE: UNIFYING VISION-LANGUAGE AND GEOMETRIC PRIORS FOR AUTONOMOUS DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

A driving planner needs both scene semantics, which vision-language models (VLMs) provide, and spatial structure, which geometric foundation models recover from visual history. We present VLG-DRIVE, which keeps both models frozen and trains only a fusion interface and a planner from trajectory labels. At each camera-grid cell, the interface fuses the paired VLM and geometry features into one joint token through a geometry-anchored residual, and a command-conditioned planner reads these tokens to predict the ego trajectory. VLG-DRIVE achieves 31.78 EPDMS on NavHard under perturbed initial states, the highest among the compared methods, and 88.27 PDMS on NavTest from cameras alone. Together with consistent gains over independent VLM and geometry streams, these results show that planning directly from the fused features of vision-language and geometric foundation models opens a new direction for building driving planners.

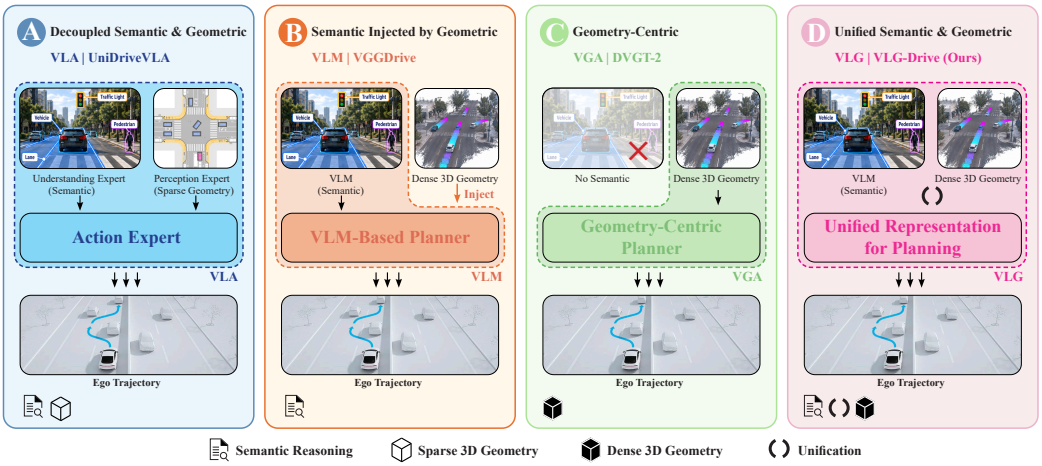


Figure 1: **Semantic–geometric interfaces for action planning.** (A) UniDriveVLA decouples semantics and geometry into an understanding expert and a sparse-geometry perception expert that feed an action expert (Li et al., 2026c). (B) VGGDrive injects dense 3D geometry into a VLM-based planner (Wang et al., 2026b). (C) DVGT-2 plans with a geometry-centric model that has no semantic stream (Zuo et al., 2026). (D) VLG-DRIVE unifies VLM semantics and dense 3D geometry in a joint representation for planning.

1 INTRODUCTION

Driving decisions depend on both what a scene means and what motion its physical structure permits. Recognizing a merging vehicle must be related to its position and the available space; interpreting a turn must account for the road layout. Vision-language models (VLMs) connect visual observations with semantic and task context (Radford et al., 2021; Liu et al., 2023; Bai et al., 2025a), while geometric foundation models learn spatial representations through reconstruction (Wang et al., 2024; 2025a). How can a driving planner make joint use of these different sources?

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Recent systems answer this question in different ways (Figure 1). UniDriveVLA decouples semantics and geometry into an understanding expert and a sparse-geometry perception expert (Li et al., 2026c), VGGDrive injects dense 3D geometry into a VLM-based planner (Wang et al., 2026b), and DVGT-2 plans with a geometry-centric model that has no semantic stream (Zuo et al., 2026). A simpler design would give a planner both feature streams and let its attention combine them. The planner must then discover, from limited trajectory labels, which VLM feature corresponds to which geometric evidence, even though both features describe the same image locations and this correspondence is known in advance. Access to both sources is therefore not sufficient; the interface that combines them matters.

Motivated by these observations, we propose VLG-DRIVE, which binds the two sources at matched camera-grid addresses before the planner reads them. The key idea is to let trajectory supervision shape how the two priors interact, so that scene meaning and spatial structure inform ego motion together. A residual interface anchored on the geometry feature combines each dense VLM–geometry pair into one joint token, which replaces the pair in the planner memory. Fusion is local while planning is global, so the interface decides how the two priors describe each location, and a separate, command-conditioned decoder decides which locations matter for the requested maneuver. Both encoders stay frozen, and trajectories are decoded by a dedicated planner rather than a language model.

We evaluate VLG-DRIVE primarily on NavHard, because its perturbed initial states require the planner to respond to a changed spatial relationship between the ego vehicle and its surroundings (Cao et al., 2025). On NavHard, VLG-DRIVE reaches 31.78 EPDMS, the highest among the compared methods. On NavTest, which evaluates recorded initial states (Dauner et al., 2024), it reaches 88.27 PDMS from camera input alone. Under matched training, pointwise fusion outperforms independent streams at every tested label fraction on both benchmarks, with the largest NavTest margin at 10% of the labels (82.23 versus 79.21 PDMS). The same interface improves a second frozen geometry backbone, DVGT-2, on both benchmarks. A component-level analysis shows that vision-language context mainly improves drivable-area compliance and lane keeping after the perturbation, whereas collision avoidance is already supported by the geometric representation.

The contributions are threefold: **(i)** VLG-DRIVE, a geometry-anchored residual interface that fuses frozen VLM and geometry features into joint tokens for a dedicated trajectory planner; **(ii)** matched comparisons showing that fusion outperforms independent streams across label fractions and geometry backbones, together with a study of which feature levels to fuse; and **(iii)** the highest NavHard EPDMS among the compared methods and strong three-camera NavTest results with frozen foundation models, together with an analysis of where vision-language context helps.

2 RELATED WORK

Vision-language models for driving and planning. End-to-end planners connect scene representations to action through bird's-eye-view aggregation (Li et al., 2022), task queries (Hu et al., 2023), vectorized scenes (Jiang et al., 2023), sensor fusion (Chitta et al., 2023), and the design of perception–planning connections (Weng et al., 2024). Foundation models add pretrained knowledge to this design space. Image-text alignment, visual instruction tuning, and the Qwen-VL family provide representations for language-conditioned scene understanding (Radford et al., 2021; Liu et al., 2023; Bai et al., 2025b;a). Flamingo and BLIP-2 learn interfaces between pretrained visual and language components (Alayrac et al., 2022; Li et al., 2023), while OpenVLA adapts pretrained visual features to robotic action (Kim et al., 2025). Driving methods built on VLMs connect these capabilities to planning in several ways. LMDrive uses language-conditioned closed-loop control (Shao et al., 2024); DriveGPT4 combines explanations and control prediction (Xu et al., 2024); DriveLM and DriveVLM organize reasoning through structured questions or hierarchical plans (Sima et al., 2024; Tian et al., 2025). EMMA represents scene elements and trajectories in a common language space (Hwang et al., 2025), and AutoVLA generates discrete trajectory actions autoregressively (Zhou et al., 2025). ReCogDrive connects VLM driving priors to a continuous diffusion planner (Li et al., 2026b). Closest to the representation question studied here, UniDriveVLA coordinates understanding, sparse perception, and action through specialized experts (Li et al., 2026c). These VLM-based planners read the scene mainly through 2D image features, which leave the metric position of semantic cues implicit. VLG-DRIVE instead combines dense geometry features with

VLM features before a dedicated trajectory decoder reads them, so that each semantic feature is tied to the 3D structure at its location and the planner knows where that evidence lies in space.

Geometric foundation models. Reconstruction provides supervision for reusable spatial features. DUSt3R predicts point maps without requiring known camera calibration or poses (Wang et al., 2024), and MASt3R adds descriptors for image matching (Leroy et al., 2024). Spann3R and CUT3R accumulate scene information sequentially (Wang & Agapito, 2025; Wang et al., 2025b); Fast3R processes multiple views jointly (Yang et al., 2025), and MonST3R addresses dynamic scenes (Zhang et al., 2025). VGGT jointly estimates cameras, depth, point maps, and tracks (Wang et al., 2025a), and VGGT-Ω extends geometric pretraining across static and dynamic scenes and studies transfer to vision-language-action models (Wang et al., 2026a). By reconstructing depth, camera poses, and point maps accurately from images alone, these models let an agent locate scene elements at concrete positions in 3D space rather than inferring their location from appearance. DVGT-2 brings this capability to driving by jointly predicting geometry and trajectories in a streaming architecture without an explicit VLM branch (Zuo et al., 2026). VGGDrive instead incorporates dense geometric features into VLM states and lets the VLM generate the trajectory (Wang et al., 2026b). VLG-DRIVE combines frozen geometric and VLM representations in a trajectory-supervised interface and plans without reconstructing an explicit scene. Unlike VGGDrive, whose trajectory is produced by the language model, VLG-DRIVE decodes the trajectory with a dedicated planner that regresses continuous ego motion in physical space instead of generating it token by token as language output, which keeps the planned trajectory stable.

Geometry priors in multimodal language models. Several recent models inject geometric features into multimodal language models. Spatial-MLLM builds on Qwen2.5-VL (Bai et al., 2025b) and merges its 2D visual features with those of a spatial encoder initialized from VGGT (Wang et al., 2025a) through a lightweight MLP connector (Wu et al., 2025). VG LLM, a video–3D geometry large language model, adds VGGT geometry features to the visual tokens of Qwen2.5-VL before the language model (Zheng et al., 2025). VLM-3R fuses reconstruction tokens from CUT3R (Wang et al., 2025b) with 2D appearance features through cross-attention for spatial and temporal reasoning (Fan et al., 2026). In all of these models, the fused features are routed back into the language model, which is tuned with instruction-style supervision and produces its output as text. Most of the VLM-based driving planners discussed above follow the same paradigm. VLG-DRIVE departs from it. It fuses geometric and vision-language features outside the language model and learns a dedicated planner directly on the fused features, which regresses continuous ego motion from trajectory labels. Neither source model is fine-tuned, and producing a plan requires no language decoding.

3 VLG-DRIVE: UNIFYING VISION-LANGUAGE AND GEOMETRIC PRIORS

3.1 PLANNING FROM COMPLEMENTARY PRIORS

Well-pretrained foundation models already capture much of what a driving planner needs. A driving-adapted VLM reads the semantics of a scene and the task it implies, and a geometric foundation model recovers the scene's spatial structure from visual history. Both have been trained at a scale that downstream driving data cannot match. Recognizing this potential, we ask whether a planner can be learned from their features alone while both foundation models stay frozen. VLG-DRIVE answers this question by unifying the two priors at the representation level.

Let $\mathcal{I}_t$ denote the current multi-camera observation, $\mathcal{H}_t$ the visual history including the current observation, u_t the route command, and m_t the recent ego-motion history included in the VLM prompt. Frozen encoders produce VLM features V_t and geometry features G_t, a trainable interface F_θ combines them into a joint planning representation Z_t, and a dedicated planner P_ϕ predicts the future trajectory:

$$\begin{aligned} V_t &= E_{\mathrm{vlm}}(\mathcal{I}_t, u_t, m_t), & G_t &= E_{\mathrm{geo}}(\mathcal{H}_t), \\ Z_t &= F_\theta(V_t, G_t), & \hat{\tau}_t &= P_\phi(Z_t, u_t). \end{aligned} \tag{1}$$

Only F_θ and P_ϕ are trained (Section 3.3). The route command conditions the VLM prompt and the planner, and motion history enters only through the prompt. This factorization separates representation formation, which decides how the two priors describe each part of the scene, from trajectory decoding, which decides which parts of the scene matter for the requested maneuver. Figure 2 summarizes the architecture.

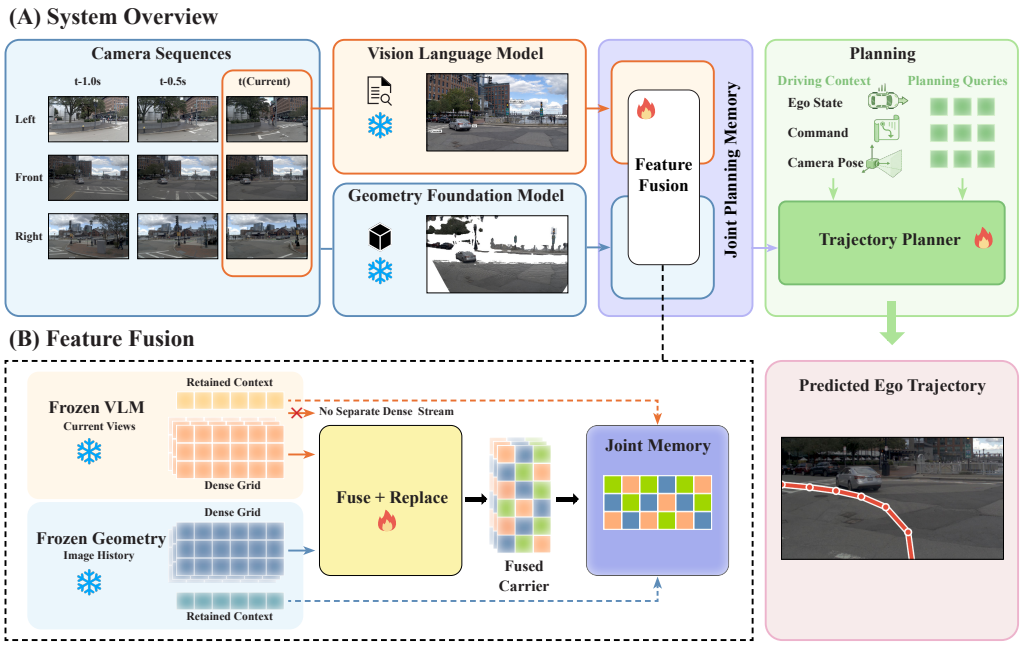


Figure 2: **The VLG-DRIVE architecture.** (A) The frozen VLM reads the current views (orange box), and the frozen geometry model reads each camera's image sequence (blue panel). Their features are fused into a joint planning memory, which the trajectory planner reads with planning queries conditioned on the route command, together with ego-state and camera-pose queries, to predict the ego trajectory. Snowflakes mark frozen modules and flames mark trainable ones. (B) At each of the $3 \times 17 \times 30$ paired cells, pointwise residual fusion adds a learned update from both sources to the geometry feature (Equation 6), and the resulting token of the fused carrier replaces both dense features in the joint memory. The retained context of each source (dashed arrows) enters the joint memory directly.

3.2 FORMING A JOINT PLANNING REPRESENTATION

Establishing correspondence. The two models are pretrained for different objectives, so matching channel dimensions alone does not make their features compatible; the camera image instead provides a common indexing domain. The interface pools both sources to a common 17×30 grid per camera and projects them to the planner width D. Each grid cell is indexed by $i = (c, h, w)$, where $c \in \{1, 2, 3\}$ selects the front-left, front, or front-right camera, and $h \in \{1, \ldots, 17\}$ and $w \in \{1, \ldots, 30\}$ are the row and column of the cell in that camera's grid. Source-specific adapters produce

$$v_i = A_v(V_i; \pi_i), \qquad g_i = A_g(G_i; \pi_i), \tag{2}$$

where V_i and G_i are the pooled features of cell i from V_t and from the current-frame grid of G_t, and π_i encodes the viewing ray of the cell. The address specifies *where* the two sources are paired, while the adapters learn *how* each source is represented for planning.

Plücker ray encoding. Grid indices alone do not tell the planner where a cell looks, because the same row and column point in different directions in different cameras. Each cell is therefore tagged with its viewing ray in the ego frame. Let $\mathbf{p}_{hw}$ be the homogeneous image coordinate of the cell center, K_c the intrinsic matrix of camera c, and R_c and $\mathbf{o}_c$ its rotation and center in the ego frame. The ray is described by its Plücker coordinates

$$\mathbf{d}_i = \frac{R_c K_c^{-1} \mathbf{p}_{hw}}{\| R_c K_c^{-1} \mathbf{p}_{hw} \|}, \qquad \mathbf{m}_i = \mathbf{o}_c \times \mathbf{d}_i, \tag{3}$$

where the unit direction $\mathbf{d}_i$ gives the orientation of the ray and the moment $\mathbf{m}_i$ distinguishes parallel rays from different camera centers. These coordinates are mapped to the planner width with fixed

random Fourier features,

$$\pi_i = \big[\sin(2\pi B^\top \mathbf{x}_i);\, \cos(2\pi B^\top \mathbf{x}_i)\big], \qquad \mathbf{x}_i = \big[\mathbf{d}_i;\, \mathbf{m}_i/s\big], \tag{4}$$

where $B \in \mathbb{R}^{6\times D/2}$ is a fixed Gaussian matrix and $s = 2\,\mathrm{m}$ scales the moment to the size of the camera rig. Both adapters add this parameter-free encoding (Appendix A), so the two sources describe each cell in the same ray coordinates.

Pointwise residual fusion. With both sources aligned at every cell, the remaining question is how to combine them. Passing the two features to the planner as separate tokens would leave it to discover, from trajectory labels alone, which VLM feature belongs to which geometric evidence. Since the grid address already specifies this correspondence, we combine each pair at its address before the planner reads the scene. The combination takes a residual form, with the geometry feature, which carries the metric structure of the cell, as the anchor and a learned update that injects vision-language context:

$$r_i = f_\theta([\mathrm{LN}_g(g_i); \mathrm{LN}_v(v_i)])\,, \tag{5}$$

$$z_i = g_i + r_i, \tag{6}$$

where f_θ is a nonlinear projection shared across locations and $[\cdot;\cdot]$ denotes channel-wise concatenation. The residual depends on both inputs and thus learns a cross-source interaction. We call the anchoring stream the *carrier*; the VLM carrier swaps the roles of g_i and v_i (Section 4.3). Zero-initializing the final projection gives $z_i = g_i$ at the start of training, so optimization begins from the geometric representation.

Fuse and replace. The complete representation is

$$Z_t = \{z_i\}_{i\in\mathcal{S}} \cup \mathcal{C}_t, \tag{7}$$

where $\mathcal{S}$ indexes the paired dense grid and $\mathcal{C}_t$ retains geometry context and global VLM tokens with no dense counterpart. The dense VLM stream thus enters the planner only through the joint tokens, which reduces the planner memory from $3{,}114$ tokens for independent streams to $1{,}584$ in the main setting.

3.3 Action-conditioned trajectory prediction

A dedicated trajectory planner translates Z_t into future ego motion. It follows query-based transformer decoding (Carion et al., 2020), but unlike object queries, which form an unordered set, each query q_k corresponds to an ordered future step k and incorporates the route command u_t. Self-attention among the queries coordinates the future steps into a consistent trajectory, and cross-attention lets each step retrieve the scene evidence it needs from Z_t. Each decoder layer $\ell \in \{1,2,3\}$ predicts incremental ego motion $\hat{\delta}_k^{(\ell)} = (\Delta x, \Delta y, \Delta\psi)$ for eight future steps at 0.5-second intervals, and the final-layer increments are accumulated to form $\hat{\tau}_t$.

Training objective. Let $\bar{\delta}_k$ denote the ground-truth increment at step k, normalized per channel with training-set statistics, and let $m_k \in \{0,1\}$ mask future steps without ground truth. The trajectory loss of decoder layer ℓ is a masked Smooth-L_1 loss on the normalized increments,

$$\mathcal{L}_{\mathrm{traj}}^{(\ell)} = \frac{1}{\sum_k m_k} \sum_{k=1}^{8} m_k \sum_{j\in\{x,y,\psi\}} \mathrm{SmoothL1}\big(\hat{\delta}_{k,j}^{(\ell)} - \bar{\delta}_{k,j}\big). \tag{8}$$

Additional ego-state, ego-pose, and per-camera queries in the decoder contribute three terms, $\mathcal{L}_{\mathrm{state}}$, $\mathcal{L}_{\mathrm{pose}}$, and $\mathcal{L}_{\mathrm{cam}}$, whose targets come from the same driving log and from the frozen geometry encoder (Appendix A). The total objective applies all terms at every decoder layer,

$$\mathcal{L} = \sum_{\ell=1}^{3} \lambda_\ell \Big(\mathcal{L}_{\mathrm{traj}}^{(\ell)} + 0.2\,\mathcal{L}_{\mathrm{state}}^{(\ell)} + 0.5\,\mathcal{L}_{\mathrm{pose}}^{(\ell)} + 0.1\,\mathcal{L}_{\mathrm{cam}}^{(\ell)}\Big), \tag{9}$$

with layer weights $(\lambda_1, \lambda_2, \lambda_3) = (0.2, 0.6, 1.0)$. Supervision therefore comes entirely from the logged ego motion and the frozen encoders' own outputs.

4 EXPERIMENTS

The experiments first test whether a planner learned from frozen priors plans well from both perturbed and recorded initial states (Section 4.2). They then ask whether the way the two sources interact matters beyond giving the planner access to both (Section 4.3), whether the benefit persists with less downstream supervision and with a different geometry backbone (Section 4.4), and which feature levels should be fused (Section 4.5). Section 4.6 finally examines where vision-language context helps the plans.

4.1 EXPERIMENTAL PROTOCOL

We train on NAVSIM v1 NavTrain and evaluate on two benchmarks. NavTest contains 12,146 scenarios scored by PDMS under the official four-second non-reactive protocol (Dauner et al., 2024); PDMS multiplies no-at-fault collision and drivable-area compliance terms with a weighted average of ego progress, time to collision, and comfort. NavHard contains 5,912 perturbed-state scenarios scored by the official combined two-stage EPDMS (Cao et al., 2025). VLG-DRIVE builds its frozen feature branches on a driving-adapted 2B Qwen3-VL model, which receives the current front-left, front, and front-right images, and on VGGT-Ω, which receives three timestamps per camera, the best setting in the history-length study of Appendix B.1. Only the interface and planner are trained, 106.1M parameters in total. The planner is a three-layer decoder of width 1024 with eight attention heads and eight planning queries, trained from scratch for 24,000 updates with batch size 128, AdamW, a learning rate of 2.5×10^{-4}, and 4,000 warmup updates on cached encoder outputs. Published baseline scores come from the cited sources, and we evaluate VGGDrive and DVGT-2 from their official checkpoints (Appendix C.1); Appendix A lists the complete recipe.

4.2 OVERALL PLANNING PERFORMANCE

Perturbed-state planning. NavHard tests whether a planner produces useful motion when the ego state departs from the recorded trajectory. Table 1 reports every component at both stages and compares VLG-DRIVE with published baselines and with the two closest alternatives to our design, VGGDrive, which feeds geometry into a VLM that generates the plan, and DVGT-2, which plans from geometry alone.

Table 1: **Planning from perturbed initial states on NavHard.** Scores are percentages; ↑ marks metrics where higher is better. Bold marks the best value in each column within each stage. S1/S2 are evaluation stages, S. the stage score, and EPDMS the official combined score. NC, DAC, DDC, TLC, EP, TTC, LK, HC, and EC denote no-at-fault collisions, drivable-area and driving-direction compliance, traffic-light compliance, ego progress, time to collision, lane keeping, history comfort, and extended comfort. Scores for LTF (Dauner et al., 2024), DiffusionDrive (Liao et al., 2025), GTRS-DP (Li et al., 2025), and GuideFlow (Liu et al., 2025) are reported by Li et al. (2026a); we evaluate VGGDrive (Wang et al., 2026b) and DVGT-2 (Zuo et al., 2026). – denotes an unavailable result.

Method	Stage	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	HC↑	EC↑	S.↑	EPDMS↑
LTF	S1	**97.3**	80.2	97.8	99.3	83.4	**96.2**	92.9	**97.8**	71.1	61.3	
	S2	79.4	69.0	85.6	98.5	83.8	76.7	47.9	**97.0**	70.6	39.2	24.40
DiffusionDrive	S1	96.8	86.0	98.8	99.3	84.0	95.8	**96.7**	97.6	**79.6**	66.7	
	S2	80.1	72.8	84.4	98.4	85.9	76.6	46.4	96.3	**72.8**	40.5	27.50
GTRS-DP	S1	94.7	78.8	96.1	99.5	83.0	94.4	92.0	97.5	72.8	–	
	S2	80.3	74.4	84.9	98.0	81.9	78.8	45.4	96.7	70.1	–	23.80
GuideFlow	S1	96.6	80.5	96.3	99.3	82.3	94.9	91.5	97.7	67.8	–	
	S2	**87.3**	**76.7**	**88.8**	**99.2**	84.3	**85.1**	**49.7**	93.1	44.5	–	27.10
VGGDrive	S1	95.4	80.9	97.6	99.1	**86.6**	94.4	92.9	**97.8**	68.9	68.3	
	S2	76.5	70.3	82.4	97.9	89.7	72.8	49.3	95.8	58.6	36.8	26.09
DVGT-2	S1	96.0	**88.4**	97.6	**100.0**	85.5	96.0	95.3	**97.8**	71.1	**76.5**	
	S2	76.9	70.1	77.5	98.7	**92.0**	72.5	47.5	82.0	38.8	35.3	28.80
VLG-DRIVE	S1	**97.3**	85.6	**99.6**	99.8	84.9	95.8	**96.7**	97.6	70.7	74.9	
	S2	82.4	71.7	85.7	98.5	89.4	79.3	47.4	95.9	55.7	**41.0**	**31.78**

VLG-DRIVE reaches 31.78 EPDMS, 2.98 points above the strongest baseline, DVGT-2. The gain comes from Stage 2, which scores the plans made after perturbation. Its stage score of 41.0 is the highest reported, driven by higher NC and TTC than DVGT-2 at a small cost in ego progress. Learning a planner directly from the fused features of frozen foundation models thus yields the most robust planning under perturbation among the compared methods.

Standard-distribution planning. NavTest evaluates the same planner from recorded initial states, measuring its driving quality under the standard distribution (Table 2).

Table 2: **System-level NAVSIM v1 NavTest planning.** Scores are percentages; ↑ marks metrics where higher is better, and bold marks the best value in each column. Cameras count physical views, and auxiliary supervision refers to downstream training. E2E, VGA, and VLA denote conventional end-to-end, geometry-based, and vision-language-action planners. Published scores are taken from the cited sources, and we evaluate VGGDrive (Appendix C.1).

Method	Family	Downstream auxiliary supervision	Cameras	LiDAR	NC↑	DAC↑	TTC↑	Comfort↑	EP↑	PDMS↑
UniAD (Hu et al., 2023)	E2E	Map & bbox	8	–	97.80	91.90	92.90	**100.00**	78.80	83.40
Latent TransFuser (Dauner et al., 2024)	E2E	Map & bbox	3	–	97.40	92.80	92.40	**100.00**	79.00	83.80
TransFuser (Chitta et al., 2023)	E2E	Map & bbox	3	✓	97.78	92.63	92.89	99.98	78.88	83.88
PARA-Drive (Weng et al., 2024)	E2E	Map & bbox	8	–	97.90	92.40	93.00	99.80	79.30	84.00
DiffusionDrive (Liao et al., 2025)	E2E	Map & bbox	3	✓	98.20	96.20	94.70	**100.00**	82.20	88.10
LFG (Strong et al., 2026)	VGA	Pseudo 3D & motion	1	–	98.20	93.70	94.40	**100.00**	79.10	85.20
DVGT-2 (Zuo et al., 2026)	VGA	Dense geometry	8	–	97.80	97.20	93.90	**100.00**	83.40	88.60
DVGT-2-NAVSIM (Zuo et al., 2026)	VGA	Dense geometry	8	–	**98.70**	**97.90**	**95.80**	**100.00**	**84.30**	**90.30**
AutoVLA (Zhou et al., 2025)	VLA	CoT reasoning	3	–	96.89	92.42	88.06	99.94	75.82	80.54
ReCogDrive (Li et al., 2026b)	VLA	Driving QA	3	–	98.30	95.10	94.30	**100.00**	81.10	86.80
VGGDrive (Wang et al., 2026b)	VLA	None	3	–	96.14	93.44	91.17	99.93	80.29	84.02
VLG-DRIVE	VLA	None	3	–	98.55	96.16	95.20	**100.00**	82.27	88.27

VLG-DRIVE reaches 88.27 PDMS from three cameras, above DiffusionDrive, which additionally uses LiDAR, and 4.25 points above VGGDrive, which generates trajectories with its VLM. The two DVGT-2 entries, at 88.60 and 90.30 PDMS, are very strong baselines, because DVGT-2 trains all parameters of its geometry backbone and planner on large-scale driving data, and DVGT-2-NAVSIM further fine-tunes the full model on NAVSIM with eight cameras. Such full-parameter driving adaptation suits NavTest, whose recorded initial states reward imitating the logged ego behavior. VLG-DRIVE instead keeps both foundation models frozen, with a geometry backbone not trained for driving, and learns only 106.1M parameters. It still comes within 0.33 PDMS of DVGT-2 and exceeds the same checkpoint by 2.98 EPDMS on NavHard, which makes a lightweight planner on frozen, general-purpose priors a promising route for driving.

4.3 HOW SHOULD PRETRAINED REPRESENTATIONS INTERACT?

The system comparison does not reveal whether the gain comes from the fusion interface or simply from giving the planner both sources. Table 3 therefore compares independent streams with the VLM and geometry carriers of Section 3.2 under the same inputs; the no-injection variants keep only the partner's global context, which separates injecting the partner from compacting the memory. The geometry-carrier fusion row is the main system, and the other rows use the base recipe (Appendix A).

Table 3: **Fusion and carrier configurations on NAVSIM v1 NavTest.** Scores are percentages; ↑ marks metrics where higher is better, and bold marks the best value in each column. DDC is driving-direction compliance; other metrics follow Table 2. Training recipes are listed in Appendix A.

Dense representation	Interaction	NC↑	DAC↑	DDC↑	TTC↑	Comfort↑	EP↑	PDMS↑
Independent streams	Decoder attention only	98.15	95.34	98.23	94.63	**100.00**	81.08	87.12
VLM carrier	No dense geometry injection	97.39	94.47	98.11	93.18	99.98	79.70	85.23
VLM carrier	Pointwise fusion	98.42	95.88	98.28	94.85	**100.00**	81.85	87.78
Geometry carrier	No dense VLM injection	98.19	95.23	98.08	94.33	99.99	81.22	86.87
Geometry carrier	Pointwise fusion	**98.55**	**96.16**	**98.30**	**95.20**	**100.00**	**82.27**	**88.27**

Fuse-and-replace keeps one dense token per cell instead of two, so its benefit could come from compaction alone. The no-injection geometry carrier rules this out, since with the same memory layout

but no VLM injection it scores 86.87 PDMS, below the 87.12 of independent streams, whereas filling that memory with joint tokens raises PDMS to 88.27, with the largest gains in DAC, TTC, and EP. For the VLM carrier, fusion likewise raises PDMS from 85.23 to 87.78, and the geometry carrier scores higher in both settings, which motivates anchoring the residual on geometry. Access to both sources is therefore not enough; they should interact before readout, at matched locations and with geometry as the anchor.

4.4 Scope of the representation effect

Table 3 establishes the benefit of fusion in the main setting. We next test whether it persists across different amounts of downstream trajectory supervision and different geometry backbones, the two axes that determine how widely the approach applies.

Across downstream supervision levels. If the joint representation makes better use of the pretrained priors, its advantage should be most visible when downstream labels are scarce. Table 4 tests this by comparing geometry alone (G), both sources as independent streams (I), and their pointwise fusion (F), each trained with the main recipe on 10%, 25%, or 100% of the NavTrain samples while the frozen encoders stay unchanged (Appendix B.4).

Table 4: **Planning with reduced downstream labels.** Columns give the share of NavTrain samples used for downstream training. The subsets are nested and consist of whole driving logs, with 7,215, 18,209, and 72,902 samples from 135, 301, and 1,152 logs. NavTest PDMS and NavHard EPDMS are percentages; ↑ marks metrics where higher is better, and bold marks the best value in each column. All arms use the main training recipe; the 100% pointwise-fusion entry is the main model reported in Tables 1 and 2.

	10%		25%		100%	
Representation	PDMS↑	EPDMS↑	PDMS↑	EPDMS↑	PDMS↑	EPDMS↑
Geometry only (G)	80.21	**21.54**	84.33	24.00	86.97	27.99
Independent streams (I)	79.21	18.96	84.62	23.53	87.27	28.43
Pointwise fusion (F)	**82.23**	21.51	**85.43**	**24.03**	**88.27**	**31.78**

Fusion scores above independent streams at every label fraction on both benchmarks. Its NavTest margin is largest at 10% of the labels (82.23 versus 79.21 PDMS), where independent streams score below geometry-only memory on both benchmarks. When downstream supervision is limited, making both sources available is therefore not sufficient, and pointwise fusion helps by providing a correspondence that the planner would otherwise need to learn. On NavHard, fusion and geometry-only memory remain close at 10% and 25%, while the full-data main model reaches 31.78 EPDMS, 3.79 points above geometry only. On NavTest, fusion thus extracts the most from the frozen priors when labels are scarce; on NavHard, its advantage over geometry alone grows with supervision.

Across geometry backbones. A useful composition should hold beyond a single geometry checkpoint. DVGT-2 is a demanding test, because both its architecture and its training data differ from those of VGGT-Ω. Table 5 asks whether adding VLM features through the same interface also helps a different frozen geometry backbone. For each backbone, the geometry-only and fused planners train our interface and decoder on the frozen encoder features with one shared recipe. Because DVGT-2 encodes undistorted images, the VLM grid is resampled into its camera frame so that both sources share grid addresses (Appendix B.4).

Fusion improves both backbones on both benchmarks, by 1.30 PDMS and 3.79 EPDMS for VGGT-Ω over its full-data geometry-only arm in Table 4, and by 0.82 and 0.91 in the matched DVGT-2 pair. The NavHard gain is four times larger with VGGT-Ω, which we attribute to its broader pretraining data. Unlike DVGT-2, which is trained on driving data, VGGT-Ω learns geometry across diverse static and dynamic scenes, so its features describe scene structure in a more general form that VLM semantics can complement. NavHard rewards exactly this semantic reading of lanes and road boundaries after a perturbation (Section 4.6). The fusion interface thus carries over to a second geometry backbone, and its benefit grows with the generality of the geometric prior.

4.5 SELECTING WHICH PRETRAINED FEATURES TO FUSE

Both foundation models expose a hierarchy of internal features, while the main system fuses only the final layer of each. Since intermediate layers retain finer spatial detail, Table 6 tests whether fusing several levels helps. CAT3 fuses early, intermediate, and final geometry taps, and DUP3 repeats the final tap through three adapters to separate depth diversity from the number of feature banks; a second comparison varies which VLM block is paired with two geometry taps.

Table 5: **Alternative geometry backbones.** NavTest PDMS and NavHard EPDMS in percent; bold marks the better value per backbone. G is geometry only and F is pointwise fusion; the VGGT-Ω F row is the main model.

Backbone	Rep.	PDMS↑	EPDMS↑
VGGT-Ω	G	86.97	27.99
	F	**88.27**	**31.78**
DVGT-2	G	86.66	22.87
	F	**87.48**	**23.78**

Table 6: **Feature-level selection.** Bold marks the best value per column within each block. Geometry taps are zero-indexed, and VLM blocks are one-indexed decoder layers (Appendix B.5).

Setting	Geo. taps	VLM	PDMS↑	EPDMS↑
Geometry depth, final-layer VLM				
Final-layer fusion	O23	Final	88.27	**31.78**
CAT3	O4/O17/O23	Final ×3	**88.31**	31.34
DUP3	O23 ×3	Final ×3	88.14	31.64
Cross-source layer pairing				
LP-LATE	O17/O23	28/28	88.09	28.06
LP-PAIR	O17/O23	14/28	**88.44**	**30.56**

Final-layer fusion, the main model, achieves the best NavHard EPDMS among the geometry-depth variants and matches the best NavTest PDMS to within 0.04 points, and CAT3 and DUP3 perform similarly, which indicates that the final layer of each model already provides the features that planning needs. Pairing an intermediate VLM block with the intermediate geometry tap raises PDMS by 0.35 and EPDMS by 2.49 over using the final VLM block for both taps. Final-layer fusion is therefore a simple and effective design, and when several levels are fused, depth-matched cross-source pairing is the more effective choice.

4.6 WHERE VISION-LANGUAGE CONTEXT HELPS

The preceding studies show that VLM features improve planning, but not which driving abilities each prior provides. We therefore compare the VGGT-Ω geometry-only planner of Table 5 with the fused main system on every NavHard component (Table 9 in Appendix B.2). The gain concentrates in Stage 2, after the ego state is perturbed, where the stage score rises by 2.25 points compared with 0.80 in Stage 1. The largest Stage-2 improvements are in drivable-area compliance ($+3.19$) and lane keeping ($+2.90$), while no-at-fault collisions are unchanged ($+0.02$). Vision-language context thus keeps the displaced planner on the road and within its lane, whereas geometry already supports collision avoidance; the two priors contribute different driving abilities, which is why combining them outperforms geometry alone. Appendix C.2 further shows that the planner reads a small set of fused cells along the driving path.

5 CONCLUSION AND LIMITATIONS

VLG-DRIVE turns frozen vision-language and geometry features into a trajectory-supervised joint planning representation. It reaches 31.78 EPDMS on NavHard, the highest among the compared methods, and 88.27 PDMS on NavTest from camera input alone. Matched comparisons show that pointwise fusion outperforms independent streams at every tested label fraction and improves two different geometry backbones on both benchmarks. Vision-language context mainly improves drivable-area compliance and lane keeping after perturbation, while the geometric representation already supports collision avoidance. Planning directly from the fused features of vision-language and geometric foundation models thus offers a new direction for building driving planners. Because VLG-DRIVE keeps both foundation models frozen and trains only a lightweight interface and planner, fully fine-tuning the geometry model on larger driving datasets, as DVGT-2 does, could further improve the results given sufficient compute. Interactive closed-loop evaluation (Karnchanachari et al., 2024) and further pretrained encoder combinations will test how broadly joint representation learning benefits driving planners.

REPRODUCIBILITY STATEMENT

The appendix identifies the models supplying vision-language and geometric priors and specifies the boundary between frozen encoders and trainable planning modules, downstream training recipes, feature layout, trajectory parameterization, causal input conventions, and external-baseline protocols. It also distinguishes published baseline values from our own baseline evaluations and describes the sampling and intervention procedures used in the spatial analysis.

AI USE STATEMENT

Generative AI tools assisted with manuscript organization and editing, experiment planning, code implementation and inspection, experiment monitoring, result checking, and figure preparation. All reported values come from saved evaluation outputs, and configuration records, checkpoint identities, and scenario coverage are retained for verification. The authors are responsible for the methodology, reported results, and final manuscript.

ETHICS STATEMENT

This work studies research-stage trajectory planning for autonomous driving. The reported metrics come from offline, non-reactive evaluation. Models may inherit biases and failure modes from their pretraining and driving datasets, including geographic, weather, traffic-participant, and sensor-configuration biases. Real-world use would require closed-loop validation, safety monitors, redundancy, and human and regulatory oversight.

REFERENCES

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: A visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 23716–23736, 2022.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Wei Cao, Marcel Hallgarten, Tianyu Li, Daniel Dauner, Xunjiang Gu, Caojun Wang, Yakov Miron, Marco Aiello, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. Pseudo-simulation for autonomous driving. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 4709–4722, 2025.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision*, pp. 213–229, 2020.

Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. TransFuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):12878–12895, 2023.

Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. NAVSIM: Data-driven non-reactive autonomous vehicle simulation and benchmarking. In *Advances in Neural Information Processing Systems*, volume 37, pp. 28706–28719, 2024.

Zhiwen Fan, Jian Zhang, Renjie Li, Junge Zhang, Runjin Chen, Hezhen Hu, Kevin Wang, Peihao Wang, Huaizhi Qu, Shijie Zhou, Dilin Wang, Zhicheng Yan, Hongyu Xu, Justin Theiss, Tianlong Chen, Jiachen Li, Zhengzhong Tu, Zhangyang Wang, and Rakesh Ranjan. VLM-3R: Vision-language models augmented with instruction-aligned 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 31054–31065, 2026.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17853–17862, 2023.

Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, Yin Zhou, James Guo, Dragomir Anguelov, and Mingxing Tan. EMMA: End-to-end multimodal model for autonomous driving. *Transactions on Machine Learning Research*, 2025.

Sarthak Jain and Byron C. Wallace. Attention is not explanation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3543–3556, 2019.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8340–8350, 2023.

Napat Karnchanachari, Dimitris Geromichalos, Kok Seang Tan, Nanxiang Li, Christopher Eriksen, Shakiba Yaghoubi, Noushin Mehdipour, Gianmarco Bernasconi, Whye Kit Fong, Yiluan Guo, and Holger Caesar. Towards learning-based planning: The nuPlan benchmark for real-world autonomous driving. In *IEEE International Conference on Robotics and Automation*, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713, 2025.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3D with MASt3R. In *Proceedings of the European Conference on Computer Vision*, pp. 71–91, 2024.

Jingyu Li, Zhe Liu, Dongnan Hu, Junjie Wu, Zipei Ma, Wenxiao Wu, Chao Han, Zhihui Hao, Zhikang Liu, Kun Zhan, Jiankang Deng, Xiatian Zhu, and Li Zhang. Metis: A generalizable and efficient world-action model for autonomous driving and urban navigation. *arXiv preprint arXiv:2606.15869*, 2026a.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19730–19742, 2023.

Yongkang Li, Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, Kun Ma, Guang Chen, Hangjun Ye, Wenyu Liu, and Xinggang Wang. ReCogDrive: A reinforced cognitive framework for end-to-end autonomous driving. In *International Conference on Learning Representations*, 2026b.

Yongkang Li, Lijun Zhou, Sixu Yan, Bencheng Liao, Tianyi Yan, Kaixin Xiong, Long Chen, Hongwei Xie, Bing Wang, Guang Chen, Hangjun Ye, Wenyu Liu, Haiyang Sun, and Xinggang Wang. UniDriveVLA: Unifying understanding, perception, and action planning for autonomous driving. *arXiv preprint arXiv:2604.02190*, 2026c.

Zhenxin Li, Wenhao Yao, Zi Wang, Xinglong Sun, Joshua Chen, Nadine Chang, Maying Shen, Zuxuan Wu, Shiyi Lan, and Jose M. Alvarez. Generalized trajectory scoring for end-to-end multimodal planning. *arXiv preprint arXiv:2506.06664*, 2025.

Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. BEVFormer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *Proceedings of the European Conference on Computer Vision*, pp. 1–18, 2022.

Bencheng Liao, Shaoyu Chen, Haoran Yin, Bo Jiang, Cheng Wang, Sixu Yan, Xinbang Zhang, Xiangyu Li, Ying Zhang, Qian Zhang, and Xinggang Wang. DiffusionDrive: Truncated diffusion model for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12037–12047, 2025.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 34892–34916, 2023.

Lin Liu, Caiyan Jia, Guanyi Yu, Ziying Song, JunQiao Li, Feiyang Jia, Peiliang Wu, Xiaoshuai Hao, and Yadan Luo. GuideFlow: Constraint-guided flow matching for planning in end-to-end autonomous driving. *arXiv preprint arXiv:2511.18729*, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763, 2021.

Hao Shao, Yuxuan Hu, Letian Wang, Guanglu Song, Steven L. Waslander, Yu Liu, and Hongsheng Li. LMDrive: Closed-loop end-to-end driving with large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15120–15130, 2024.

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 *Proceedings of the European Conference on Computer Vision*, pp. 256–274, 2024.

Matthew Strong, Wei-Jer Chang, Quentin Herau, Jiezhi Yang, Yihan Hu, Chensheng Peng, and Wei Zhan. Learning to drive is a free gift: Large-scale label-free autonomy pretraining from unposed in-the-wild videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 32144–32153, 2026.

Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, XianPeng Lang, and Hang Zhao. DriveVLM: The convergence of autonomous driving and large vision-language models. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4698–4726, 2025.

Hengyi Wang and Lourdes Agapito. 3D reconstruction with spatial memory. In *International Conference on 3D Vision*, pp. 78–89, 2025.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5294–5306, 2025a.

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-Ω. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026a.

Jie Wang, Guang Li, Zhijian Huang, Chenxu Dang, Hangjun Ye, Yahong Han, and Long Chen. VGGDrive: Empowering vision-language models with cross-view geometric grounding for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10954–10964, 2026b.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3D perception model with persistent state. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10510–10522, 2025b.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20697–20709, 2024.

Xinshuo Weng, Boris Ivanovic, Yan Wang, Yue Wang, and Marco Pavone. PARA-Drive: Parallelized architecture for real-time autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15449–15458, 2024.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pp. 11–20, 2019.

Diankun Wu, Fangfu Liu, Yi-Hsin Hung, and Yueqi Duan. Spatial-MLLM: Boosting MLLM capabilities in visual-based spatial intelligence. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kwan-Yee K. Wong, Zhenguo Li, and Hengshuang Zhao. DriveGPT4: Interpretable end-to-end autonomous driving via large language model. *IEEE Robotics and Automation Letters*, 9(10):8186–8193, 2024.

Jianing Yang, Alexander Sax, Kevin J. Liang, Mikael Henaff, Hao Tang, Ang Cao, Joyce Chai, Franziska Meier, and Matt Feiszli. Fast3R: Towards 3D reconstruction of 1000+ images in one forward pass. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21924–21935, 2025.

Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. MonST3R: A simple approach for estimating geometry in the presence of motion. In *International Conference on Learning Representations*, 2025.

Duo Zheng, Shijia Huang, Yanyang Li, and Liwei Wang. Learning from videos for 3D world: Enhancing MLLMs with 3D vision geometry priors. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

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. In *Advances in Neural Information Processing Systems*, volume 38, pp. 27920–27956, 2025.

Sicheng Zuo, Zixun Xie, Wenzhao Zheng, Shaoqing Xu, Fang Li, Hanbing Li, Long Chen, Zhi-Xin Yang, and Jiwen Lu. DVGT-2: Vision-geometry-action model for autonomous driving at scale. *arXiv preprint arXiv:2604.00813*, 2026.

A Implementation Details

Sources of vision-language and geometric priors. The vision-language branch uses a driving-adapted Qwen3-VL model with 2B parameters. The geometry branch uses VGGT-Ω. Both encoders are frozen during downstream training. VLM features correspond to the current front-left, front, and front-right images. The main geometry features are extracted per camera with visual history, and the current-frame grid is passed to the planner.

Dense layout and spatial indexing. Both branches supply a 17×30 grid per camera. Pooling and camera ordering define corresponding row-major addresses, so correspondence is defined at the grid level. Source-specific adapters project the features to the decoder width. Ray directions are computed at cell centers from the recorded intrinsics and camera-to-ego extrinsics, and the Plücker encoding of Equation 4 is applied to both sources. The geometry adapter adds the encoding to its projected feature. The VLM adapter uses a second encoding of the same form with its own random matrix, maps it through a zero-initialized linear layer, and adds it to V_i before projection, so the VLM features are unchanged at initialization. The geometry adapter additionally adds a learned camera embedding and a fixed two-dimensional sinusoidal encoding of the grid position.

Fusion and memory. The pointwise fusion module is a two-layer MLP with GELU activation, source-specific LayerNorm, and a zero-initialized output layer. Concatenating independent streams retains $1{,}530$ dense VLM tokens and $1{,}530$ dense geometry tokens, together with context, for $3{,}114$ memory tokens. Pointwise fusion retains $1{,}530$ joint tokens, three camera tokens, 48 register tokens, a null token, and two global VLM tokens, totaling $1{,}584$ decoder memory tokens.

Training configurations. The main system and the data-efficiency, backbone, feature-level, and history-length studies use the main recipe; Appendix B.3 lists the recipe of each fusion-study row. Table 7 compares the two recipes. Both have three decoder layers, eight attention heads, and a feed-forward expansion of four. The main recipe has 106.1M trainable parameters outside the frozen encoders. When the source representation is fixed, frozen-encoder outputs are cached once and reused during downstream training; experiments that vary internal feature levels extract the relevant features online.

Table 7: **Downstream training recipes.** Both keep the pretrained encoders frozen and train the downstream modules from scratch.

Setting	Base recipe	Main recipe
Decoder width	512	1024
Decoder layers	3	3
Training iterations	12,000	24,000
Batch size	256	128
Learning rate	10^{-3}	2.5×10^{-4}
Warmup iterations	1,000	4,000

Planning queries and losses. Eight planning queries represent future poses at 0.5-second intervals over four seconds. A learned four-way command embedding (left, straight, right, or unknown) conditions these queries. Each decoder layer applies query self-attention, memory cross-attention, and a feed-forward block. The model predicts normalized per-step $(\Delta x, \Delta y, \Delta\psi)$ in the NAVSIM ego frame and is trained with the objective of Equation 9. Outputs are denormalized per channel and cumulatively summed. The ego-state, ego-pose, and per-camera queries share the decoder layers with the planning queries, and their prediction heads serve only the training objective. $\mathcal{L}_{\text{state}}$ is an L_1 loss on the normalized current velocity and acceleration. $\mathcal{L}_{\text{pose}}$ is a Smooth-L_1 loss on the ego motion since the previous frame, with the heading change represented by its sine and cosine. $\mathcal{L}_{\text{cam}}$ is a Smooth-L_1 consistency loss between the predicted camera-pose encodings and those of the frozen geometry encoder. All three targets are derived from the logged ego motion and the model inputs.

Prompt and causal inputs. The VLM prompt contains the native NAVSIM command and a recent history-trajectory string in the convention expected by the driving model. Motion history reaches the model only through this prompt and the geometry history.

B Additional Representation Studies

B.1 Geometry history length

Table 8: **Temporal context in the three-camera fusion planner.** The vision-language branch receives the current views; only the number of timestamps supplied per camera to the geometry model changes. Frozen encoders, full-data pointwise fusion, the decoder, and the 24,000-update main recipe are shared across rows. The three-timestamp row is the main system of Tables 1 and 2. Scores use the percentage scale; ↑ marks metrics where higher is better, and bold marks the best value in each column.

Geometry timestamps	NavTest PDMS↑	NavHard EPDMS↑
1	83.30	28.68
2	87.85	30.68
3	**88.27**	**31.78**
4	**88.27**	30.52

Table 8 shows that visual history is essential for the geometry branch. A second timestamp raises NavTest PDMS from 83.30 to 87.85 and NavHard EPDMS from 28.68 to 30.68. A third timestamp raises both scores further, to 88.27 PDMS and 31.78 EPDMS, the best NavHard result. A fourth timestamp leaves NavTest at 88.27 and lowers NavHard to 30.52. Three timestamps therefore give the best overall result, which is the setting of the main system. Each row is evaluated on all 12,146 NavTest and 5,912 NavHard scenarios with the official two-stage combined score.

Why longer history stops helping. The first additional timestamp supplies parallax for recovering scene depth and the recent motion of the ego vehicle and surrounding agents, neither of which a single frame provides. It accounts for most of the benefit, 4.55 PDMS and 2.01 EPDMS. The third timestamp extends this motion context and still adds 0.42 PDMS and 1.09 EPDMS. Beyond three, the oldest observation carries little new information about the current decision, while the frozen geometry encoder must compress a longer sequence into the same current-frame grid that the planner reads. NavTest starts from recorded states that continue the observed motion, and its score is unchanged by the fourth timestamp. NavHard's second stage starts from states displaced from the recorded trajectory, where the decision depends mainly on the current layout around the displaced ego vehicle rather than on the past motion; there the fourth timestamp lowers EPDMS by 1.26 points.

B.2 NavHard components with and without vision-language context

Table 9 lists every NavHard component for the VGGT-Ω geometry-only planner of Table 5 and the fused main system, the two planners compared in Section 4.6. The table values are the official aggregate component scores of each stage.

Table 9: **NavHard components with and without vision-language context.** Geometry only (G) is the VGGT-Ω geometry-only planner of Table 5; pointwise fusion (F) is the main system of Tables 1 and 2. Scores are percentages; ↑ marks metrics where higher is better, and bold marks the better value within each stage. Metric abbreviations follow Table 1.

Stage	Representation	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	HC↑	EC↑	S.↑
S1	Geometry only (G)	**97.78**	84.44	99.33	**100.00**	84.15	**96.22**	96.22	**97.78**	68.00	74.06
	Pointwise fusion (F)	97.33	**85.56**	**99.56**	99.78	**84.92**	95.78	**96.67**	97.56	**70.67**	**74.87**
S2	Geometry only (G)	82.42	68.56	85.25	98.40	87.76	77.88	44.50	**96.65**	**56.04**	38.74
	Pointwise fusion (F)	**82.43**	**71.74**	**85.70**	**98.47**	**89.39**	**79.31**	**47.40**	95.87	55.68	**40.99**

B.3 Fusion-study controls

Table 3 uses shared three-camera inputs and frozen encoder settings. The no-injection geometry carrier removes the paired dense VLM grid but keeps global VLM context. The no-injection VLM

carrier removes dense geometry but keeps geometry camera and register context. Pointwise fusion combines the two features at a matched grid address. The pointwise geometry carrier is the main system of Table 2, trained with the main recipe; the other rows use the base recipe, and the two VLM-carrier rows form a matched pair. All component values and PDMS are scenario-weighted means over the same $12{,}146$ valid NavTest scenarios.

B.4 Data-efficiency and backbone-transfer protocols

Reduced downstream labels. The 10% and 25% subsets are nested whole-log subsets of NavTrain with fixed membership, so every scenario of a smaller subset also belongs to the larger ones. Trajectory normalization statistics are computed from the training data of each fraction. All arms use the main recipe with $24{,}000$ updates at batch size 128, so smaller subsets are revisited proportionally more often. The geometry-only arm (G) disables the VLM pathway, the independent-stream arm (I) passes both dense grids to the decoder memory, and the fusion arm (F) uses pointwise fusion. The full-data fusion entry and the VGGT-Ω fusion entry in Table 5 both reuse the main model reported in Tables 1 and 2. Each arm is evaluated on all $12{,}146$ NavTest and $5{,}912$ NavHard scenarios.

Second geometry backbone. The DVGT-2 arms use the released frozen DVGT-2 encoder features computed from undistorted images. The VLM dense grid, extracted from the raw images, is transformed into the undistorted camera frame by calibration-derived bilinear feature resampling, and the same corrected intrinsics define the ray descriptors of both sources. The two global VLM tokens are kept unchanged. The geometry-only and fused DVGT-2 planners each train a new downstream head on the full $72{,}902$-example training set with the main recipe.

B.5 Feature-level controls

Shared downstream recipe. Table 6 uses three cameras, three-frame geometry histories, a current-frame 17×30 grid per camera, and a three-layer, width-1024 decoder. Each configuration trains the adapters, fusion interface, and planner from scratch for 24,000 updates with batch size 128, learning rate 2.5×10^{-4}, and 4,000 warmup steps. CAT3/DUP3 and LP-LATE/LP-PAIR share the training sample order within each pair. All configurations use the training objective of Equation 9, and the encoders remain frozen. Each reported configuration covers 12,146 unique valid NavTest scenarios and 5,912 unique valid NavHard scenarios.

Geometry-depth control. The final-layer fusion row reuses the main model of Tables 1 and 2, which fuses final-layer geometry and VLM features into one bank. The controlled geometry-depth comparison is CAT3 versus DUP3. O4, O17, and O23 denote zero-indexed geometry taps. CAT3 uses an independent geometry adapter for each tap; DUP3 applies three independent adapters to the same raw O23 feature. A shared VLM projection and shared fusion MLP serve all banks. All fused banks enter the shared key/value memory of every decoder layer. Context is retained once. For three banks, dense attention logits receive a $-\log 3$ bias to control the increased dense-versus-context multiplicity. CAT3 and DUP3 match trainable parameters and memory size.

Cross-source layer-pairing control. LP-LATE and LP-PAIR extract both frozen representations for each sample. The VLM receives 960×544 current-view images, with the same command and causal history prompt in both arms. Its selected states are image-token outputs of language-decoder blocks 14 and 28. Block 14 is passed through the frozen final RMSNorm; block 28 is already normalized. Geometry uses O17 and O23, where O17 is a deep intermediate tap. LP-LATE uses block 28 for both banks, whereas LP-PAIR uses block 14 with O17 and block 28 with O23. The arms share the training sample order, geometry inputs, independent per-bank geometry adapters, VLM projection, fusion MLP, and decoder recipe. Both banks enter every decoder layer's main memory, with a $-\log 2$ dense-logit bias. Late-layer geometry context and global VLM tokens are retained once in both arms.

C Evaluation and Diagnostic Protocols

C.1 External baselines and aggregation

Published NavTest rows follow the comparison in Wang et al. (2026b); AutoVLA and ReCogDrive use its supervised-fine-tuning setting. Camera counts in Table 2 denote distinct physical views consumed by each model before any image stitching. The NAVSIM reproductions of UniAD and PARA-Drive use all eight views, whereas Latent TransFuser and TransFuser use the three front-facing views (Dauner et al., 2024); TransFuser additionally uses LiDAR. DiffusionDrive retains the TransFuser camera and LiDAR setup (Liao et al., 2025). LFG uses a single monocular front camera (Strong et al., 2026). AutoVLA, ReCogDrive, and VGGDrive use the front-left, front, and front-right views (Zhou et al., 2025; Li et al., 2026b; Wang et al., 2026b). The DVGT-2 NavTest row reports the official result of Zuo et al. (2026) for its released checkpoint with eight views, the same checkpoint that we evaluate on NavHard. The DVGT-2-NAVSIM row reports their official result after fine-tuning DVGT-2 on NAVSIM with eight views and four frames.

All our baseline evaluations use the official checkpoints released by the authors, without additional training or fine-tuning. For VGGDrive, we run the official NAVSIM checkpoint and inference code on both NavTest and NavHard, with a causal prompt that contains only information available at inference time. For DVGT-2, we evaluate the official base checkpoint on NavHard with eight views, four frames, and official image undistortion.

For external baselines predicting only (x, y), headings are constructed with a forward tangent at non-terminal points and a consistent terminal convention. The NavHard result is read from the official `extended_pdm_score_combined` summary, which combines the two stages at the scenario level before averaging.

C.2 Attention and spatial interventions

Where the planner reads. Planning queries place 99.94% of their cross-attention on the fused dense grid, which holds 96.6% of the memory tokens. Within the grid, attention concentrates on the region the vehicle will traverse, as a projected reference-path tube occupies 2.19% of the dense cells but receives 12.89% of the attention mass. Splitting the dense attention by camera and averaging within each command group shows that the reading follows the route command. Left-command scenes place 36.9% of the attention on the front-left camera and 23.6% on the front-right camera; right-command scenes reverse this ordering, with 15.1% and 32.8%. Straight-command scenes concentrate 59.5% of the attention on the front camera, and scenes without a known command spread it nearly evenly across the three views. The same fused memory is therefore read differently according to the requested maneuver.

Attended locations drive the prediction. To test whether this attention reflects reliance (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019), we replace selected features with the mean dense feature of the same camera in every decoder layer and recompute the trajectories. Replacing the top 5% of attended cells shifts waypoints by 0.717 m, compared with 0.066 m for equal-size random cells and 0.0009 m for the bottom 5%. Replacing the path tube itself shifts waypoints by 0.414 m, compared with 0.043 m for a count-matched random control. The planner's predictions therefore depend on a small set of localized memory cells concentrated along the driving path.

Protocol. The analysis uses the VLG-Drive model reported in Tables 1 and 2. The diagnostic set is drawn from the 3,373 validation samples by taking, within each command group (left, straight, right, and unknown), one scene from each of the low, middle, and high trajectory-error strata, so that it spans every route command and a range of difficulty.

Cross-attention is recovered for all decoder layers, attention heads, and planning queries by replaying the observed attention calls with their masks; the recomputed outputs match the original calls to within 1.91×10^{-5}. Quantitative measurements use the native attention grid.

In every decoder layer, selected dense key/value inputs are replaced by the mean dense feature from the same camera. The top and bottom 5% of attended cells are compared with three random selections of the same size. A second intervention targets a one-cell-radius neighborhood of valid, visible

projected reference waypoints, using count-matched random controls. Projection uses $(x, y, 0)$ and calibrated camera transforms, since the trajectory's third channel is heading. Invisible reference nodes are omitted, and the reference trajectory serves only to define the diagnostic region.

The reported shift is the mean distance between the perturbed and original predicted waypoints. Averaged over scenes, the path tube receives $6.74\times$ its share of cells in attention mass.