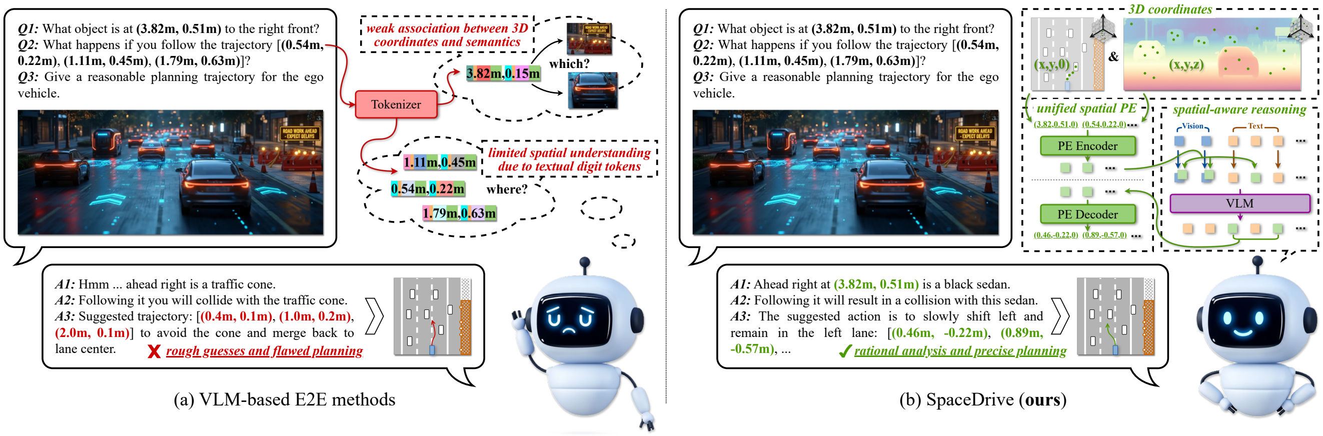
SpaceDrive: Infusing Spatial Awareness into VLM-based Autonomous Driving

Peizheng Li* [1,2], Zhenghao Zhang* [1,4], David Holtz[1], Hang Yu[1,5], Yutong Yang[1,6],
Yuzhi Lai[2], Rui Song[7], Andreas Geiger[2,3], Andreas Zell[2]
[1]Mercedes-Benz AG, [2]University of Tübingen, [3]Tübingen AI Center,
[4]TU Munich, [5]Karlsruhe Institute of Technology, [6]University of Stuttgart, [7]UCLA

https://zhenghao2519.github.io/SpaceDrive_Page/



Q1: What object is at **(3.82m, 0.51m)** to the right front?
Q2: What happens if you follow the trajectory [**(0.54m, 0.22m), (1.11m, 0.45m), (1.79m, 0.63m)**]?
Q3: Give a reasonable planning trajectory for the ego vehicle.

A1: Hmm ... ahead right is a traffic cone.
A2: Following it you will collide with the traffic cone.
A3: Suggested trajectory: [**(0.4m, 0.1m)**, **(1.0m, 0.2m)**, **(2.0m, 0.1m)**] to avoid the cone and merge back to lane center.

(a) VLM-based E2E methods

Q1: What object is at **(3.82m, 0.51m)** to the right front?
Q2: What happens if you follow the trajectory [**(0.54m, 0.22m), (1.11m, 0.45m), (1.79m, 0.63m)**]?
Q3: Give a reasonable planning trajectory for the ego vehicle.

A1: Ahead right at **(3.82m, 0.51m)** is a black sedan.
A2: Following it will result in a collision with this sedan.
A3: The suggested action is to slowly shift left and remain in the left lane: [**(0.46m, -0.22m)**, **(0.89m, -0.57m)**, ...

(b) SpaceDrive (**ours**)

Figure 1. **Spatial awareness in VLM-based end-to-end autonomous driving.** (a) Constrained by insufficient 3D pre-training and discrete token-wise encoding, existing end-to-end planners based on the VLM struggle to precisely ground, associate, and predict 3D spatial positions, limiting their planning capabilities. (b) Our proposed SpaceDrive planner introduces a unified 3D coordinate encoding to replace the original VLM's textual digit tokens and augment visual features, achieving explicit association with 2D perspective semantics to enhance joint spatial reasoning for E2E planning. Compared to current VLM-based methods, it achieves state-of-the-art driving capability in the nuScenes open-loop evaluation and the second-best driving performance in the Bench2Drive closed-loop simulation.

Abstract

End-to-end autonomous driving methods built on vision language models (VLMs) have undergone rapid development driven by their universal visual understanding and strong reasoning capabilities obtained from the large-scale pre-training. However, we find that current VLMs struggle to understand fine-grained 3D spatial relationships which is a fundamental requirement for systems interacting with the physical world. To address this issue, we propose SpaceDrive, a spatial-aware VLM-based driving framework that treats spatial information as explicit positional encodings (PEs) instead of textual digit tokens, enabling joint reasoning over semantic and spatial representations. SpaceDrive employs a universal positional encoder to all 3D coordinates derived from multi-view depth estimation, historical ego-states, and text prompts. These 3D PEs are first superimposed to augment the corresponding 2D visual tokens. Meanwhile, they serve as a task-agnostic coordinate representation, replacing the digit-wise numerical tokens as both inputs and outputs for the VLM. This mechanism enables the model to better index specific visual semantics in spatial reasoning and directly regress trajectory coordinates rather than generating digit-by-digit, thereby enhancing planning accuracy. Extensive experiments validate that SpaceDrive achieves state-of-the-art open-loop performance on the nuScenes dataset and the second-best Driving Score of 78.02 on the Bench2Drive closed-loop benchmark over existing VLM-based methods.

1. Introduction

Large-scale pre-trained VLMs are known for their vast knowledge bases and strong reasoning capabilities. Leveraging VLMs to assist [28, 48, 59] or replace [12, 54, 61] traditional end-to-end (E2E) autonomous driving (AD) sys-

[*] Equal contribution, names are sorted alphabetically.
Correspondence to: peizheng.li@mercedes-benz.com.

tems has therefore emerged as a prominent trend recently. These systems typically reformulate AD functions into natural language, and flexibly perform scene understanding, motion prediction and trajectory planning based on semantic information extracted from images. Compared to fixed modular designs [19, 27], VLM-based E2E models promise to achieve superior generalization, addressing increasingly complex and dynamic driving scenarios.

However, current VLMs demonstrate clear limitations in 3D tasks such as geometric measurement and distance estimation [5, 65, 67], which are critical for autonomous driving. This issue stems mainly from two primary factors, as illustrated in Fig. 1.a. First, the absence of 3D-data-based pre-training forces models to rely on inference from existing 2D knowledge. When dealing with 3D coordinates, VLMs struggle to associate them with the corresponding objects and their 2D semantics, leading to ambiguous or even incorrect scene descriptions [78]. Second, language models inherently treat numerical processing as digit-by-digit classification. This classification overlooks the inherent inter-digit proximity between numerical tokens and incorrectly averages the importance of different token positions [11].

In autonomous driving, existing VLM-based planners either introduce task-specific embeddings tailored to individual downstream tasks [12, 50] or represent waypoints as sequences of numeric tokens directly generated by the language model [54, 61]. The former relies on specialized 3D fine-tuning, tying embeddings to particular tasks and domains and thus hindering a transferable, universal spatial representation that preserves VLM generalization. The latter suffers from the aforementioned limitations in the numerical modeling ability of language models, results in inaccurate waypoint predictions. However, an important but under-emphasized aspect is that the Transformer architecture is **inherently capable of processing positional relationships between tokens**, which can be conceptualized as **spatial relationships between semantic features** [29]. Therefore, extending this capability to 3D spatial awareness becomes a natural and logical idea.

Inspired by this, we propose *SpaceDrive*, a spatial-aware VLM-based AD framework illustrated in Fig. 1.b, which incorporates a universal encoding for 3D positions to enhance spatial understanding and reasoning in VLMs. Specifically, we first encode 3D coordinates derived from depth estimation and add them onto corresponding 2D visual tokens, establishing an explicit association between semantic features and 3D spatial locations. Meanwhile, this 3D PE serves as a general coordinate representation, replacing either conventional coordinates in natural language or task-specific embeddings as the input and output of VLM. Furthermore, for the output PE, we replace the original classification-based design with regression-based decoder and loss to address the numerical prediction deficiencies in language models. Our framework also exhibits strong adaptability to various VLM base models and reasoning strategies, further underscoring its potential as a universal paradigm.

To directly validate the trajectory planning accuracy, we first conducted an open-loop evaluation. Experiments on the nuScenes dataset [4] demonstrate that SpaceDrive achieves state-of-the-art performance among all VLM-based methods. However, similarity-based open-loop planning evaluation is highly susceptible to dataset overfitting, offering only limited insight into the model's actual driving competence. Therefore, we further validate our method on the closed-loop Bench2Drive [25] benchmark where we achieve a Driving Score of 78.02 (second-best in VLM-based planners), further confirming its capability to perform reasonable planning in dynamic and complex scenarios.

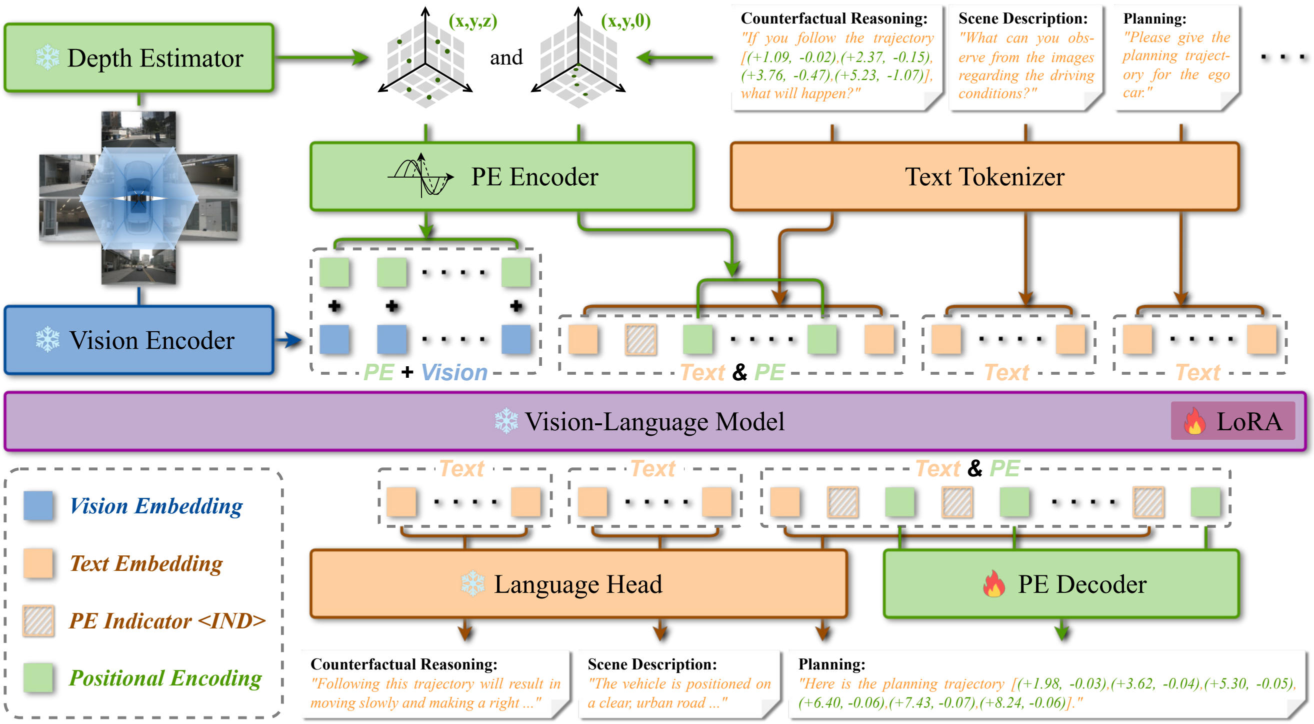
The contributions of this paper are as follows:

- We identify fundamental limitations of current VLMs in 3D spatial reasoning and waypoint prediction, and propose *SpaceDrive*, a spatial-aware VLM-based AD framework with a universal 3D positional encoding that explicitly associates image semantics with 3D coordinates.
- *SpaceDrive* employs a shared 3D PE as a general coordinate representation to augment visual tokens and serve as the coordinate interface for language models, along with a regression-based decoder to enhance the end-to-end trajectory planning.
- Our framework achieves state-of-the-art performance in open-loop planning on nuScenes, while exhibits strong closed-loop planning capabilities under complex driving scenarios on the Bench2Drive benchmark.

2. Related Work

End-to-End Autonomous Driving Over the past years, end-to-end autonomous driving has evolved from traditional modular stacks [9, 33, 34, 37, 45, 62, 71, 76] to fully differentiable, planning-oriented designs. After early methods like ST-P3 [18] achieved joint optimization of perception and planning, UniAD [19] unified the entire stack into a query-based framework, using planning supervision to regularize upstream tasks. Building on this paradigm, follow-up studies [6, 24, 27, 53, 64, 80] achieved further improvements in planning efficiency and decision quality. A key inflection came from AD-MLP [74] and BEV-Planner [38], which exposed open-loop brittleness: simple ego-state priors can rival sophisticated stacks. This finding shifted attention toward closed-loop fidelity and benchmarks that align with driving quality, *e.g.* Bench2Drive [25] and DriveE2E [72], and stimulated numerous subsequent methods [20, 24, 35, 39, 40, 43, 51, 75, 84] based on them. Despite their strong performance, conventional E2E frameworks lack generalized scene understanding, thus struggling to handle complex and dynamic driving scenarios.

Spatial Intelligence of VLMs Recently, spatial intelligence



Counterfactual Reasoning: "If you follow the trajectory [(+1.09, -0.02),(+2.37, -0.15),(+3.76, -0.47),(+5.23, -1.07)], what will happen?"

Scene Description: "What can you observe from the images regarding the driving conditions?"

Planning: "Please give the planning trajectory for the ego car."

Counterfactual Reasoning: "Following this trajectory will result in moving slowly and making a right ..."

Scene Description: "The vehicle is positioned on a clear, urban road ..."

Planning: "Here is the planning trajectory [(+1.98, -0.03),(+3.62, -0.04),(+5.30, -0.05),(+6.40, -0.06),(+7.43, -0.07),(+8.24, -0.06)]."

Figure 2. **SpaceDrive framework.** Beyond the base VLM, a frozen depth estimator predicts dense metric depths from surround-view images, which are projected into 3D coordinates and encoded by a universal PE encoder to augment visual tokens with spatial cues. BEV coordinates in text prompts are encoded by the same PE encoder, replacing the original coordinate tokens and preceded by the PE indicator ⟨IND⟩. At the output stage, the recognized PE is passed through a PE decoder to obtain the final coordinates for trajectory planning.

in VLMs has progressed from 2D relational heuristics to explicit 3D-aware reasoning [5, 13, 31, 32, 65, 79, 83]. This trend was initiated by SpatialVLM [5], which synthesized large-scale spatial Visual Question Answering (VQA) data to support both qualitative and quantitative spatial reasoning from 2D images. Subsequent works injected 3D structure more directly into the modeling pipeline. From integration of 3D features and positional embeddings in Scene-LLM [13] and LLaVA-3D [83], to dynamic and region-prompted spatial reasoning in Video-3D LLM [79], Spatial-MLLM [65], and SR-3D [8], these works collectively advance language-guided 3D understanding, grounding, and planning. Besides, dedicated benchmarks have standardized the evaluation. VSI-Bench [67] probes egocentric video-based visual–spatial intelligence with more than 5,000 QA pairs, while STI-Bench [36] stresses precise spatial–temporal estimation (pose, displacement, motion) across various scene setting. These studies demonstrate the immense potential of VLMs in spatial-aware tasks and suggest clear benefits for the perception, prediction, and planning in autonomous driving.

VLMs-Based Driving Agents Vision-language and multimodal LLMs have reshaped E2E driving by injecting priors, interactivity, and explicit reasoning into perception-prediction-planning. Early work such as DriveGPT4 [66] formulated driving as a language-conditioned sequence modeling, pairing video inputs with textual rationales to produce interpretable low-level controls. VLP [48] and DriveVLM [59] extended this direction by leveraging large vision-language models for scene understanding and trajectory generation, while DriveLM [54] further strengthened structured reasoning via graph-structured VQA over driving scenes. Recent methods [73, 81, 82] achieve further enhancements in areas such as reinforcement learning, symbolic reasoning, and precise control. For example, OmniDrive [61] pursues holistic 3D grounding with counterfactual supervision, while ORION [12] aligns reasoning and action spaces via a long-horizon QT-Former, an LLM reasoner, and a generative planner for strong closed-loop scores. Concomitant with the methodological developments, corresponding benchmarks [1, 47, 52, 54, 61] have also arisen, primarily targeting on open-world reasoning and regulation compliance. Nevertheless, existing VLM-based autonomous driving systems suffer from an inadequate treatment of 3D spatial awareness, a critical deficiency that forms the core focus of this paper.

3. Method

As illustrated in Fig. 2, we propose SpaceDrive, a spatial-aware framework that enhances end-to-end planning through explicit injection of 3D information into the VLM architecture. Specifically, the surrounding images are first encoded by a visual encoder, and then aligned to the language model's semantic space via a projector. Meanwhile, these images are processed by a depth estimator to obtain absolute depths, which are converted into 3D positional encodings through a universal PE encoder. The visual tokens and their 3D PEs are then added element-wise, yielding spatially-aware visual tokens that serve as inputs to the VLM. Besides, text prompts for various reasoning tasks are also fed into the VLM as text token inputs. Notably, Bird's-Eye-View (BEV) or 3D coordinates within these prompts are processed separately by the same PE encoder to generate universal PEs, replacing the corresponding original text tokens. To avoid semantic confusion with other tokens, a predefined PE indicator is placed before each PE input and output. During reasoning, these PEs leverage their intrinsic similarity for direct interaction and indexing of the spatially-aware visual tokens. At the output stage, general textual outputs are decoded by the language head, while coordinate-related outputs are recognized and decoded by a dedicated PE decoder to produce accurate 3D coordinates for precise trajectory planning.

3.1. Spatial Awareness in Perception

A prerequisite for spatial intelligence is reliable 3D scene understanding, *i.e.* establishing dense correlations between 2D perspective visual features and their 3D geometry.

Vision Encoding A pretrained vision encoder $f_{vis.}$ first converts the K multi-view images $\{I_k\}_{k=1}^{K}$ into N patch tokens:

$$X_v = f_{vis.}(\{I_k\}) = \{x_p\}_{p=1}^{N}. \tag{1}$$

Given that our primary goal is the explicit infusion of spatial awareness, the sparse and highly abstract features within Q-Former-style architectures [61] are fundamentally limited in directly associating with concrete 3D spatial locations. Furthermore, the efficacy of the Q-Former typically requires additional large-scale pre-training for vision-language alignment, largely reducing the adaptabilty of our framework. Therefore, we keep using a simple MLP g to densely align the visual and language feature spaces, consistent with general-purpose VLMs [2, 41]:

$$H_v = g(X_v) = \{h_p\}_{p=1}^{N}. \tag{2}$$

Spatial Encoding To obtain 3D scene information, a pretrained depth estimator $f_{dep.}$ produces dense per-view absolute depth maps $D_k = f_{dep.}(I_k)$. To prioritize the foreground, for each patch p with image-plane support $\mathcal{R}_p$ we assign the minimum depth $d_p = \min_{(u,v)\in\mathcal{R}_p} D_k(u,v)$ as its corresponding depth. With the per-camera calibration matrix $\mathcal{P}_k$, we project the patch center (u_p, v_p) to 3D as $\mathbf{c}_p = \mathcal{P}_k^{\dagger}[u_p, v_p, d_p, 1]z\top$ to obtain explicit metric coordinates. Each $\mathbf{c}_p = (x_p^{3D}, y_p^{3D}, z_p^{3D})$ is then encoded into a universal 3D positional encoding via a PE encoder. To minimize confusion with the existing RoPE [56] used in the VLM, we opt for a 3D sine-cosine positional encoding extending the standard 1D formulation dimension-wise:

$$\begin{gathered}\phi(\mathbf{c}_p) = \left[\phi_x(x_p^{3D}), \phi_y(y_p^{3D}), \phi_z(z_p^{3D})\right] \in \mathbb{R}^{dim}, \text{with} \\ \phi_a(p_a) = \begin{cases}\sin(\frac{p_a}{20000^{2i/d_a}}), \\ \cos(\frac{p_a}{20000^{2i/d_a}}),\end{cases} \quad i = 0, \ldots, \lfloor \frac{d_a}{2} \rfloor - 1, \\ d_x = d_y = \lceil \frac{dim}{3} \rceil, d_z = dim - d_x - d_y.\end{gathered} \tag{3}$$

for spatial dimension $a \in \{x, y, z\}$ and total PE width dim.

Spatial Token Injection Prior works [12, 61] inject learnable 3D cues within or before the vision-language projector, yielding only implicit geometry. In contrast, we explicitly add metric 3D coordinates information $\phi(\mathbf{c}_p)$ on top of modality-aligned visual tokens h_p after the MLP g. This design enables later reuse of the same PE $\phi(\cdot)$ for coordinates from text prompts, allowing the model to directly index spatially grounded visual features and strengthening downstream spatial reasoning, as further discussed in Sec. 3.2.

It is worth noting that direct additive injection of $\phi(\mathbf{c}_p)$ shifts the token norm distribution away from the pretrained VLM regime. To mitigate this, we introduce a learnable normalization factor α_{PE} shared across all 3D PEs, simply

$$\tilde{H_v} = \{\tilde{h_p}\}_{p=1}^{N}, \ \tilde{h}_p = h_p + \alpha_{PE}\, \phi(\mathbf{c}_p). \tag{4}$$

3.2. Spatial Awareness in Reasoning

Existing VLMs exhibit strong general 2D multimodal reasoning yet remain deficient in explicit 3D spatial inference:

1. Insufficient pretraining on metric 3D data and spatial reasoning tasks confines current VLMs mainly to abstract 2D reasoning [83], yielding poor estimation of inter-object spatial relations, physical extent, and distances.
2. The classification-based numerical prediction in existing language models often prioritizes fitting data distributions while neglecting the inherent affinity between numerical symbols and their sequential order [11], thereby degrading precision in continuous waypoint predictions.

Alternatively, existing methods introduce task-specific queries and decode explicit 3D coordinates from them using MLPs [50], generative modules [12] or attention layers [83]. Although partially mitigating the above limitations, the resulting tokens lack unified spatial semantics and thus transfer poorly across tasks. In contrast, we reuse the previously defined 3D PE $\phi(\mathbf{c})$ as a universal spatial representation. This choice enforces representational consistency between perception and reasoning, improving accuracy of coordinate handling and estimation within the VLM.

Encoding of Coordinates in Text Prompts During tokenization of input text prompts, we scan the text sequence $\{t_i\}_{i=1}^{L}$ for substrings $\mathcal{S}$ expressing spatial coordinates. For each detected coordinate expression we extract its numeric values as a vector $\mathbf{c}_r = (x_r, y_r, z_r)$. The same 3D positional encoder $\phi(\cdot)$ as in Sec. 3.1 is then applied to obtain a corresponding spatial token $\phi(\mathbf{c}_r) \in \mathbb{R}^{dim}$, which replaces the original sequence of numeric tokens corresponding to that coordinate. Each input PE is preceded by a specifically defined token, ⟨IND⟩, serving as the PE identifier (for simplicity, ⟨IND⟩ will be omitted in subsequent descriptions and formulations). The adjusted text token inputs are as follows:

$$\tilde{H}_t = \{\tilde{h}_i\}_{i=1}^{L},\ \tilde{h}_i = \begin{cases} \phi(\mathbf{c}_r) & i \in \mathcal{S}_r \\ \text{Tokenizer}(t_i) & \text{otherwise} \end{cases}. \quad (5)$$

A special case arises for BEV coordinates (*e.g.* trajectory waypoints), where we set all z-axis components in the PE $\phi(\mathbf{c}_r)$ to 0 so that they do not contribute to subsequent attention calculations.

Encoding of the Ego Status It has been verified that ego state inputs are highly effective for trajectory planning [38, 74]. Existing approaches typically encode all state variables (*e.g.* pose, velocity, acceleration) simply into a single vector embedding $\mathbf{e}_{\text{ego}} \in \mathbb{R}^{dim}$, mostly also augmented with BEV features to obscure explicit metric structure. Thanks to our unified spatial representation, we instead encode the historical ego waypoints via the same $\phi(\cdot)$ employed before, *i.e.* $\{\phi(\mathbf{c}_\tau^{ego})\}_{\tau=-T}^{1}$. It will then be fed into the language model together with $\mathbf{e}_{\text{ego}}$ as explicit spatial-temporal conditioning for accuracy trajectory planning.

Decoding of Text with Coordinates At the output stage, the VLM produces a sequence of embeddings $\{\mathbf{e}_j\}_{j=1}^{J}$. A standard language head W_{lang} maps each $\mathbf{e}_j$ to a distribution over the textual vocabulary $\mathcal{V}$ for ordinary decoding. Additionally, we utilize the previously defined ⟨IND⟩ to signal a forthcoming coordinate emission, extending the original W_{lang} to W'_{lang}, *i.e.*

$$y_j = \arg\max_{y \in \mathcal{V}'} \left(W'_{\text{lang}}\mathbf{e}_j\right)_y,\ \mathcal{V}' = \mathcal{V} \cup \{\langle\text{IND}\rangle\}. \quad (6)$$

If $y_j \neq \langle\text{IND}\rangle$ the text token is emitted normally. When $y_j = \langle\text{IND}\rangle$, $\mathbf{e}_j$ remains in the language context and the subsequent output state $\mathbf{e}_{j+1}$ is routed to a PE decoder $\psi(\cdot)$ to produce metric coordinates:

$$\hat{\mathbf{c}} = \psi(\mathbf{e}_{j+1}),\ \hat{\mathbf{c}} \in \mathbb{R}^3 \quad (7)$$

This mechanism yields precise BEV trajectory waypoints (omitting the z-coordinate) while preserving autoregressive continuity for surrounding text. Because the composite sinusoidal encoding $\phi(\mathbf{c})$ is not analytically invertible (phase and frequency aliasing across dimensions), $\psi(\cdot)$ is set as a fully learnable MLP and trained to regress ground-truth coordinates. The shared use of $\phi(\cdot)$ in both perception and reasoning ensures that $\psi(\cdot)$ operates over embeddings already aligned with unified spatial PEs, improving coordinate fidelity and trajectory planning accuracy.

3.3. Loss Function

A typical training objective combines language modeling $\mathcal{L}_{\text{LM}}$ (applied to all text outputs) with coordinate regression $\mathcal{L}_{\text{reg.}}$ (applied to all coordinate outputs, such as waypoints):

$$\mathcal{L} = \mathcal{L}_{\text{LM}} + \mathcal{L}_{\text{reg.}}(\hat{\mathbf{c}}, \mathbf{c}), \quad (8)$$

where $\mathcal{L}_{\text{reg.}}$ may vary with the type of the decoder $\psi(\cdot)$ and the adopted trajectory generation strategy. For the basic MLP decoder, we adopt the Huber loss for coordinate regression.

4. Experiments

4.1. Experimental Settings

Dataset and Metrics The nuScenes dataset [4] comprises 1,000 urban driving scenes (train/val/test: 700/150/150) with full-stack 360° sensing (6 cameras, 1 LiDAR, 5 radars). For open-loop planning we predict 6 waypoints within a 3 s horizon and evaluate (i) waypoint displacement (L2) error, (ii) Collision rate (fraction of future timestamps overlapping with any dynamic agent), and (iii) Intersection rate (fraction of timestamps intruding into non-drivable map regions).

Bench2Drive [25] is a closed-loop planning benchmark emphasizing interactive scenarios (merging, overtaking, yielding, emergency negotiation) in a deterministic CARLA V2 [10] simulator. Our closed-loop evaluation adopts the official protocol of 220 short routes, covering 44 interactive scenarios, with 5 distinct routes defined for each scenario. Closed-loop metrics include Driving Score (route progress penalized by safety infractions) and Success Rate (percentage of scenarios completed without terminal violation). All reported results adopt identical horizon, temporal sampling, footprint inflation, and map definitions for fair comparison.

Implementation Details Our model adopts Qwen2.5-VL-7B [2] as the base VLM. We finetune the core LLM using LoRA [17] with the rank of 16, while keeping the original vision encoder and vision-language projector frozen. Unidepthv2-ViT-L [49] is chosen as our default depth estimation module without additional finetuning. For the open-loop evaluation on nuScenes, the model is trained for 6 epochs on 8×A100 80GB GPUs with a batch size of 8. The learning rate is set to 1e-4 and cosine annealing is used to ensure stable training. The input resolution is resized to 640 × 640. For the closed-loop evaluation, the model is trained for 12 epochs using the same training setup as the open-loop evaluation. For VQA training and evaluation, we adopt the data and settings utilized in OmniDrive [61]. Further details are provided in the supplementary materials.

4.2. Quantitative Results

Open-loop Planning To directly validate the impact of spatial awareness on VLM's coordinate regression, we first conducted the open-loop planning evaluation on the nuScenes

Table 1. **Open-loop planning results on nuScenes [4].** SpaceDrive+ denotes the adoption of the ego planner input. Methods categorized as Hybrid Paradigm simultaneously stack traditional and VLM-based approaches, and are thus incomparable. Results are highlighted in **bold** and underline for the best and second-best performance among VLM-based methods, respectively. All results in this table follow the evaluation protocol of OmniDrive [61] and ORION [12]. More methods based on other metrics are provided in the supplementary materials.

Method	Ego Status		L2 (m) ↓				Collision (%) ↓				Intersection (%) ↓			
	BEV	Planner	1s	2s	3s	Avg.	1s	2s	3s	Avg.	1s	2s	3s	Avg.
Traditional Modular Paradigm														
ST-P3 [18]	-	-	1.59	2.64	3.73	2.65	0.69	3.62	8.39	4.23	2.53	8.17	14.40	8.37
UniAD [19]	✓	✓	0.20	0.42	0.75	0.46	0.02	0.25	0.84	0.37	0.20	1.33	3.24	1.59
VAD-Base [27]	✓	✓	0.17	0.34	0.60	0.37	0.04	0.27	0.67	0.33	0.21	2.13	5.06	2.47
AD-MLP [74]	-	✓	0.15	0.32	0.59	0.35	0.00	0.27	0.85	0.37	0.27	2.52	6.60	2.93
BEV-Planner++ [38]	✓	✓	0.16	0.32	0.57	0.35	0.00	0.29	0.73	0.34	0.35	2.62	6.51	3.16
UAD [14]	✓	✓	0.13	0.28	0.48	0.30	0.00	0.19	0.61	0.27	0.13	1.08	2.89	1.37
VLM-based Paradigm														
EMMA [22]	-	-	**0.14**	**0.29**	0.54	**0.32**	-	-	-	-	-	-	-	-
RDA-Driver [21]	✓	✓	0.23	0.73	1.54	0.80	**0.00**	**0.13**	0.83	0.32	-	-	-	-
DriveVLM [59]	-	✓	0.18	0.34	0.68	0.40	0.10	0.22	**0.45**	0.27	-	-	-	-
ORION [12]	✓	-	0.17	0.31	0.55	0.34	0.05	0.25	0.80	0.37	-	-	-	-
OmniDrive-Q [61]	-	-	1.15	1.96	2.84	1.98	0.80	3.12	7.46	3.79	1.66	3.86	8.26	4.59
OmniDrive-Q++ [61]	✓	✓	**0.14**	**0.29**	0.55	0.33	**0.00**	**0.13**	0.78	0.30	0.56	2.48	5.96	3.00
SpaceDrive (**ours**)	-	-	1.06	1.79	2.55	1.80	0.35	1.33	3.97	1.88	0.96	3.38	8.28	4.21
SpaceDrive+ (**ours**)	-	✓	0.15	**0.29**	**0.51**	**0.32**	0.04	0.18	0.49	**0.23**	**0.22**	**0.80**	**2.79**	**1.27**
Hybrid Paradigm														
VLP [48]	✓	-	0.30	0.53	0.84	0.55	0.01	0.07	0.38	0.15	-	-	-	-
ReAL-AD [46]	✓	-	0.30	0.48	0.67	0.48	0.07	0.10	0.28	0.15	-	-	-	-
DriveVLM-Dual [59]	✓	-	0.15	0.29	0.48	0.31	0.05	0.08	0.17	0.10	-	-	-	-
SOLVE-VLM [7]	✓	-	0.13	0.25	0.47	0.28	0.00	0.16	0.43	0.20	-	-	-	-
Senna [28]	✓	-	0.11	0.21	0.35	0.22	0.04	0.08	0.13	0.08	-	-	-	-

Table 2. **Closed-loop planning results on Bench2Drive [25].** Results are highlighted in **bold** and underline for the best and second-best performance among VLM-based methods, respectively.

Method	Closed-loop Metric	
	Driving Score ↑	Success Rate(%) ↑
Traditional Modular Paradigm		
AD-MLP [74]	18.05	0.00
UniAD-Base [19]	45.81	16.36
VAD-Base [27]	42.35	15.00
MomAD [55]	44.54	16.71
GenAD [80]	44.81	15.90
SparseDrive [57]	47.38	17.72
UAD [14]	49.22	20.45
WoTE [35]	61.71	31.36
ThinkTwice [24]	62.44	37.17
DriveTransformer-L [26]	63.46	38.60
DriveAdapter [23]	64.22	42.08
HiP-AD [58]	86.77	69.09
VLM-based Paradigm		
ReAL-AD [46]	41.17	11.36
Dual-AEB [77]	45.23	10.00
X-Driver [44]	51.70	18.10
GEMINUS [60]	65.39	37.73
VDRive [15]	66.25	50.51
StuckSolver [3]	70.89	50.01
DriveMoE [69]	74.22	48.64
ETA [16]	74.33	48.33
VLR-Drive [30]	75.01	50.00
ORION [12]	77.74	54.62
SimLingo [50]	**85.07**	**67.27**
SpaceDrive+ (**ours**)	78.02	55.11

dataset. As shown in Tab. 1, SpaceDrive+ achieves the SOTA performance across all reported metrics, consistently surpassing existing VLM-based methods. The lowest L2 error (0.32) indicates that coordinate-level regression allows closer adherence to expert driving trajectories. Simultaneously, the markedly reduced Collision (0.23%) and Intersection (1.27%) rates further show that SpaceDrive+ excels not only at fitting ground truth but also enhances autonomous driving safety comprehensively through superior spatial understanding and reasoning.

Notably, our method does not include the BEV features widely adopted in existing pipelines. This provides evidence that a unified positional encoding is sufficient for 3D spatial modeling within VLM-oriented autonomous driving, obviating dense BEV representations. Considering the sensitivity of open-loop metrics to the integration of ego status [38], we further report the variant without ego status inputs, *i.e.* SpaceDrive. In this setting, our method also surpasses its base model (OmniDrive [61]) across all dimensions (L2: -0.18, Collision: -1.91%, Intersection: -0.38%), validating the effectiveness of explicitly injecting 3D spatial information.

Closed-loop Planning In Tab. 2 we conduct closed-loop evaluation to establish a comprehensive and reliable assessment of planning performance. While our base model (OmniDrive) attains competitive open-loop metrics, its text-only planning paradigm fails drastically in closed-loop simulation (Under 10% Success Rate). Empirically, its predicted trajectories collapse into near-linear paths with unstable heading

Table 3. **Ablation of positional encoding.** Here, $\phi(\mathbf{c}_r)$, $\phi(\mathbf{c}_p)$ and $\phi(\mathbf{c}_t^{ego})$ denote the usage of spatial positional encodings for textual coordinate inputs, 3D coordinates corresponding to vision tokens and past ego locations, respectively.

Exp.	$\phi(\mathbf{c}_p)$	$\phi(\mathbf{c}_r)$	$\phi(\mathbf{c}_\tau^{ego})$	Avg. L2 ↓	Avg. Col. ↓	Avg. Int. ↓
SpaceDrive						
1				2.51	4.53	6.77
2	✓			1.88	2.45	2.36
3		✓		2.42	5.06	8.94
4	✓	✓		1.80	1.88	4.21
SpaceDrive+						
5				0.41	0.60	4.40
6	✓	✓		0.33	0.23	1.32
7	✓	✓	✓	0.32	0.23	1.27

oscillations. This substantiates our hypothesis that pure natural-language trajectory generation primarily fits data priors rather than learning a controllable driving pattern. More comparisons are provided in the supplementary materials.

By introducing explicit spatial tokens, SpaceDrive+ achieves 78.02 Driving Score and 55.11% Success Rate, ranking as the second-best VLM-based method (notably, SimLingo [50] employs extensive data augmentation via Action Dreaming). These gains indicate that injecting structured 3D positional information is sufficient to unlock strong closed-loop planning within a VLM-oriented framework.

4.3. Qualitative Results

Figure 3 illustrates a representative Bench2Drive scenario in which the ego vehicle is required to avoid collision with two cyclists ahead. The planner first accelerates to probe the opportunity for overtaking and lane changing. After observing that the adjacent vehicle does not yield, our spatial-aware SpaceDrive+ detects sufficient rearward clearance in the target lane and opts to decelerate to create a safe insertion gap. Once the opening emerges, it executes a decisive lateral maneuver. As the lane change nears completion, the model infers from its ego state and surrounding vehicle positions that rapid heading re-alignment is necessary to avoid drifting out of lane boundaries. This case exemplifies that the injected 3D spatial encoding enables SpaceDrive+ to adapt its strategy to evolving scene geometry and generate safety-aware plans.

4.4. Ablation Studies

Our approach focuses on endowing models with spatial awareness rather than tailoring them for open- or closed-loop planning. Therefore, considering that closed-loop performance may be influenced by factors such as training strategies, PID controller tuning, and other pipeline heuristics, we conducted our ablations under open-loop settings (excluding the ego status to prevent overfitting) to ensure the reliable and fair comparisons.

Positional Encoding We compare the effect of injecting explicit positional encodings into different modules of the VLM-based planner in Tab. 3. First, adding spatial encoding to vision tokens (Exp. 2 vs. 1) yields substantial improvements on all metrics, *i.e.*-0.63 L2, -2.08% Collision and -4.14% Intersection. This is largely attributable to the enhanced spatial understanding achieved by supplying 3D geometric context alongside 2D image features. Meanwhile, the gains from replacing the textual coordinate with PE, as demonstrated in Exp. 3, are relatively smaller, likely because the PE lacks the bridge to associate with 2D semantic space in pretrained VLMs. However, when a unified positional encoding is applied to both vision and textual coordinate streams (Exp. 4 vs. 1; Exp. 6 vs. 5), planning performance improves regardless of the use of ego status, underscoring the value of a shared spatial representation. Finally, with ego status enabled, injecting past ego positions using the same $\phi(\cdot)$ (Exp. 7 vs. 6) further reduces L2 error and Intersection rates. This indicates that the benefits coming from consistent spatial tokens are stable and reliable, facilitating spatial understanding and reasoning in VLMs.

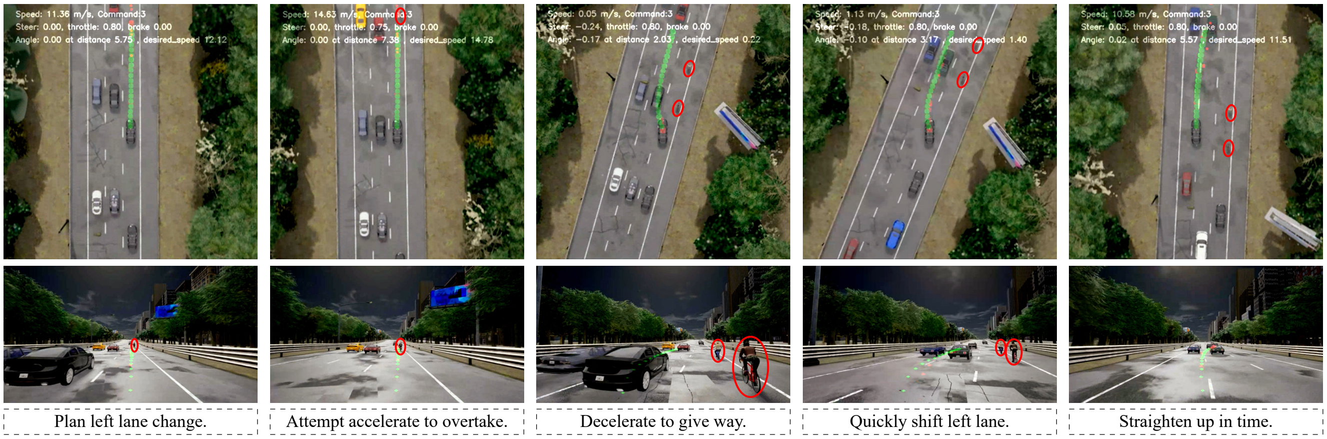
Table 4. **Ablation of PE encoder & decoder.** Gray indicates that only 4929 out of 5119 output samples are semantically reasonable.

Exp.	Encoder $\phi(\cdot)$	Decoder $\psi(\cdot)$	Avg. L2 ↓	Avg. Col. ↓	Avg. Int. ↓
4	Sine-Cosine	Coordinate-wise	1.80	1.88	4.21
8	MLP	Coordinate-wise	1.96	3.17	6.76
9	RoPE	Coordinate-wise	1.93	3.71	11.40
10	Sine-Cosine	Sine-Cosine	1.87	2.62	9.20
11	Sine-Cosine	Task-specific	1.93	2.41	5.58

Table 5. **Ablation of PE normalization.** Gray indicates that only 2421 out of 5119 output samples are semantically reasonable.

Init. α_{PE}	Learnable	Avg. L2 ↓	Avg. Collision ↓	Avg. Intersection ↓
1		2.34	3.63	8.46
0.1		2.43	3.79	9.42
0.02		2.22	2.71	10.17
1	✓	1.82	2.04	4.62
0.1	✓	1.80	1.88	4.21
0.02	✓	1.86	2.03	5.42

PE Encoder & Decoder Table 4 compares different encoders and decoders of PE. Using a Sine–Cosine encoder yields a translation-invariant encodability. It assists the attention layers in recovering inter-token spatial relations, giving clear gains over a fully learnable MLP encoder (Exp. 4 vs. 8). Although RoPE shares the same property as the additive Sine–Cosine encoding, it leads to a large performance degradation and instability due to the confusion with existing RoPE used in the base VLM (Exp. 4 vs. 9). On the decoder side, numerically inverting Sine–Cosine encodings to precise coordinates is ill-posed (only coarse interpolation is possible), and the output embedding space of a large VLM is typically not fully aligned with its input space. These factors make a learnable coordinate-wise MLP decoder preferable, as reflected in its lower L2 error (1.80 in Exp. 4 vs 1.87 in Exp. 10). A common paradigm in VLM planners is to use a


Speed: 11.36 m/s, Command:3
Steer: 0.00, throttle: 0.80, brake 0.00
Angle: 0.00 at distance 5.75 , desired_speed 12.12
Speed: 14.63 m/s, Command:3
Steer: 0.00, throttle: 0.75, brake 0.00
Angle: 0.00 at distance 7.38 , desired_speed 14.78
Speed: 0.05 m/s, Command:3
Steer: −0.24, throttle: 0.80, brake 0.00
Angle: −0.17 at distance 2.03 , desired_speed 0.22
Speed: 1.13 m/s, Command:3
Steer: −0.18, throttle: 0.80, brake 0.00
Angle: −0.10 at distance 3.17 , desired_speed 1.40
Speed: 10.58 m/s, Command:3
Steer: 0.05, throttle: 0.80, brake 0.00
Angle: 0.02 at distance 5.57 , desired_speed 11.51

Plan left lane change.

Attempt accelerate to overtake.

Decelerate to give way.

Quickly shift left lane.

Straighten up in time.

Figure 3. **Qualitative results of closed-loop evaluation on Bench2Drive [25].** Green and pink dots represent path and speed waypoints, respectively. Red circles indicate cyclists ahead that the vehicle needs to avoid. Parameters such as speed and steering wheel angle can be found in the figures.

Table 6. **SpaceDrive based on different VLM foundations.**

Method	Base VLM	Avg. L2 ↓	Avg. Collision ↓	Avg. Intersection ↓
SpaceDrive	LLaVA	1.82	2.44	4.08
	Qwen-VL	1.80	1.88	4.21
SpaceDrive+	LLaVA	0.31	0.23	1.42
	Qwen-VL	0.32	0.23	1.27

task-specific embedding and decode an entire trajectory from it. This limits reuse across tasks and forces retraining when objectives change. The comparison between experiments 4 and 11 shows that jointly decoding multiple waypoints from a single embedding underperforms the coordinate-wise strategy in all metrics, which predicts each waypoint conditioned on shared spatial tokens.

PE Normalization In transformer-based VLMs, the embedding norm directly modulates its relative importance in the attention operations. Consequently, the norm of the PE can largely affect training stability and planning accuracy. In Tab. 5 we compare performance under different fixed initialization scales α_{PE}. For Qwen-VL, smaller static α_{PE} values lead to consistent degradation across all open-loop metrics and even semantic instability, implying that excessively small PE norms result in negligible attention scores and hinder convergence. Since the optimal PE scale differs between foundation models and may shift over training, we promote α_{PE} to a learnable parameter. This adaptive normalization, while mitigating semantic instability, produces a about -0.5 m reduction in Avg. L2 together with marked moderation in Collision and Intersection rates. This outcome validates the importance of a learnable normalization coefficient for the unified 3D PE.

More ablations and experiments regarding VQA, depth estimator, etc., are provided in the supplementary materials.

4.5. Adaptability

Our proposed 3D spatial representation also demonstrates excellent adaptability. First, it attains comparable performance on both Qwen-VL and LLaVA (See Tab. 6), indicating that the strong performance arise from unified spatial reasoning rather than backbone-specific biases. Injecting spatial awareness also preserves compatibility with inference-time reasoning enhancements. For example, without ego state inputs we observe distributional degeneration of predicted trajectories, *i.e.* mode collapse. Augmenting SpaceDrive with lightweight chain-of-thought prompting (CoT) stabilizes waypoint diversity and reduces collapse without retraining. Furthermore, we can also adapt the PE decoder into a VAE-based generative model, which has also been proven effective in improving closed-loop robustness [12]. More comparisons are provided in the supplementary materials. All these results collectively show that our spatial encoding is a general booster for spatial reasoning and E2E planning across VLM foundations and inference paradigms.

5. Conclusion

In this paper, we presented SpaceDrive, a spatial-aware VLM-based end-to-end autonomous driving framework, that unifies 3D spatial awareness and multimodal reasoning through a universal positional encoding interface. The approach infuses metric 3D positional encodings into visual tokens, textual coordinate mentions, and historical ego states, and decodes coordinates with a regression head instead of digit-wise language generation. In this way, the model achieves superior trajectory planning performance compared to methods relying solely on natural language fitting. Besides, this design reduces reliance on task-specific embeddings, unleashing the generalization capabilities of pre-trained VLMs for space-relevant reasoning. Extensive experiments show state-of-the-art open-loop planning results on nuScenes across displacement, collision, and map compliance metrics, as well as strong closed-loop performance on Bench2Drive, substantially narrowing the gap between VLM planners and specialized end-to-end baselines. Ablations confirm the benefit of unified spatial tokens, coordinate-wise decoding, and learnable normalization of positional encodings. Meanwhile, SpaceDrive also achieves good transferability across VLM backbones and remains adaptable to language reasoning strategies. Limitations include the absence of explicit uncertainty handling, and no exploitation of multi-frame temporal memory mechanisms, which also represent potential directions for future research. Nevertheless, we believe the proposed unified spatial representation offers a principled path toward more reliable, generalizable spatial-aware VLM-driven autonomy.

Acknowledgement

This work is a result of the joint research project STADT:up (Förderkennzeichen 19A22006O). The project is supported by the German Federal Ministry for Economic Affairs and Climate Action (BMWK), based on a decision of the German Bundestag. The author is solely responsible for the content of this publication.