Think Densely, Act Sparsely: Latent Expert Cognitive Chains for Vision-Language-Action Autonomous Driving

Jie Wang[1,2*] **Guang Li**[3] **Zhijian Huang**[3] **Jinlong Li**[3] **Chenxu Dang**[3]
Hangjun Ye[3] **Yahong Han**[1†] **Long Chen**[3]

[1]School of Artificial Intelligence, Tianjin University
[2]School of Artificial Intelligence, Shanxi University [3]Xiaomi EV

{wangjiexy, yahong}@tju.edu.cn, lguang@live.cn, zhijianh11@gmail.com,
jinlong.szu@gmail.com, {dangchenxu, yehangjun, chenlong37}@xiaomi.com

Project Page: https://wj-cv.github.io/LECDrive/

Abstract

Autonomous driving fundamentally demands dense understanding of visual, semantic, and geometric information, yet existing Vision-Language-Action (VLA) models are typically trained with sparse supervision such as language instructions or trajectory signals. This direct mapping from dense visual observations to sparse representations forces the model to compress or even discard critical scene information essential for safe driving. We argue that this limitation does not stem from insufficient model capacity, but from the lack of explicit mechanisms that encourage the formation of dense world representations in the latent space. In this spirit, we propose a novel paradigm for driving intelligence—**Think Densely, Act Sparsely**—and instantiate it with an efficient and interpretable framework, **LECDrive** (**L**atent **E**xpert Cognitive **C**hains for Vision-Language-Action autonomous **Driv**ing). The core idea is to mimic the hierarchical perception process of humans by embedding a progressive chain of latent expert cognition within the VLA model. Concretely, we inject a compact set of task-specific latent tokens as carriers, enabling the model to internalize dense knowledge from foundational vision models, including visual representations (DINOv3), semantic structures (SAM3), and spatial geometry (DepthAnything3), while maintaining efficient inference. This interdependent cognitive chain follows a curriculum learning principle, where supervision is progressively structured from representation to semantics and geometry. Extensive experiments demonstrate that LECDrive consistently improves the performance of base VLMs across multiple autonomous driving benchmarks, covering key tasks such as object perception, state prediction, and trajectory planning. These findings suggest that constructing dense world models through structured latent expert chains is a promising direction for VLA-based autonomous driving.

1 Introduction

Autonomous driving is a decision-making task that critically depends on dense world understanding [1–3], involving fine-grained visual appearance, comprehensive semantic structure, and precise spatial geometry [4, 5]. These components are not independent but are hierarchically abstracted and integrated during human perception, making autonomous driving fundamentally a dense world modeling problem [6, 7]. Recent advances in large-scale Vision-Language Models (VLMs) [8–10] have introduced new opportunities for addressing long-tail scenario understanding and complex semantic reasoning in autonomous driving [11–13]. Building on this, unified Vision-Language-Action (VLA) frameworks [14–16] adopt end-to-end learning to map multimodal inputs and instructions

*Work done while interning at Xiaomi Embodied Intelligence Team.
†Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

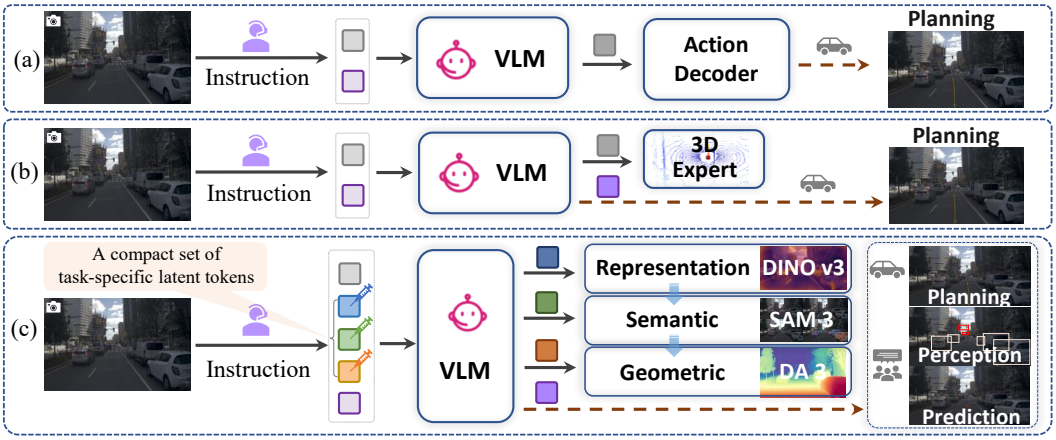


Figure 1: Mainstream paradigms vs. LECDrive. (a) Decoupled modeling [15, 16, 22]. (b) Geometry-enhanced modeling [24–26]. (c) Our Think Densely, Act Sparsely paradigm with latent expert chains, enabling progressive dense world modeling via a compact set of task-specific latent tokens.

directly to driving trajectories in an autoregressive manner [14, 17, 18]. However, the reliance on task-specific sparse supervision, such as language instructions or trajectory waypoints, results in a significant compression of dense visual information during training. This limitation hinders the model's ability to fully capture complex scene structures and ultimately constrains both scene understanding and trajectory planning performance [14, 19–21].

Consequently, some approaches resort to a decoupled modeling paradigm (Figure 1(a)) by introducing an additional action decoder for trajectory prediction [15, 16, 22, 23]. Although this design improves planning accuracy, it decouples scene understanding from decision making, limiting the transfer of the VLM's perception and reasoning capabilities to downstream planning. In parallel, to compensate for the inherent limitations of VLMs in spatial geometric reasoning, other methods adopt explicit geometry-enhanced pipelines (Figure 1(b)). These approaches [4, 24–26] incorporate 3D expert features or introduce auxiliary tasks such as point cloud or occupancy prediction on top of the visual hidden states. Although such strategies improve geometric awareness, they suffer from two key limitations. First, explicit integration of 3D features typically incurs substantial inference overhead [25]. Second, naively stacking multiple geometry-related objectives often leads to task interference, hindering representation learning and resulting in suboptimal solutions [24, 26]. More critically, these methods [24–26] primarily focus on geometric augmentation and fail to address the fundamental issue of dense visual information loss in VLA models. In essence, autonomous driving requires not only enhanced geometric reasoning but also rich visual representations and coherent semantic structures to support robust decision making. However, this requirement is systematically overlooked in existing approaches [15, 16, 22–26], where dense pixel-level information is prematurely compressed into sparse action representations. We argue that this limitation does not stem from insufficient model capacity, but from the absence of explicit mechanisms that enforce, or at least encourage, the construction of structured dense world representations in the latent space.

In this spirit, we propose a new modeling paradigm for autonomous driving—**Think Densely, Act Sparsely**. The central hypothesis is that if a VLA model is guided to explicitly construct dense world representations in its latent space, progressing through a cognitive chain from visual representation to semantic understanding and finally to spatial geometry, it can achieve more accurate and robust driving decisions while maintaining efficient sparse action outputs. To instantiate this idea, we introduce **LECDrive** (**L**atent **E**xpert Cognitive **C**hains for Vision-Language-Action **Driv**ing), an efficient and interpretable latent expert chain framework (Figure 1(c)). LECDrive injects a compact set of task-specific latent tokens into the VLM, forming a progressive latent expert chain with three hierarchical stages. This design enables the model to gradually activate and internalize dense knowledge from foundational vision models, significantly enhancing its representational capacity with minimal impact on inference cost. The proposed chain follows a curriculum learning principle that mirrors the hierarchical nature of human perception. First, DINOv3 [27] provides feature-level supervision to establish robust visual representations. Second, SAM3 [28] introduces semantic segmentation signals to guide the learning of traffic-relevant semantic structures. Finally, DepthAnything3 [29] supplies geometric supervision to enhance spatial geometric understanding. These stages do not constitute independent multi-task objectives; instead, they form an interdependent cognitive chain

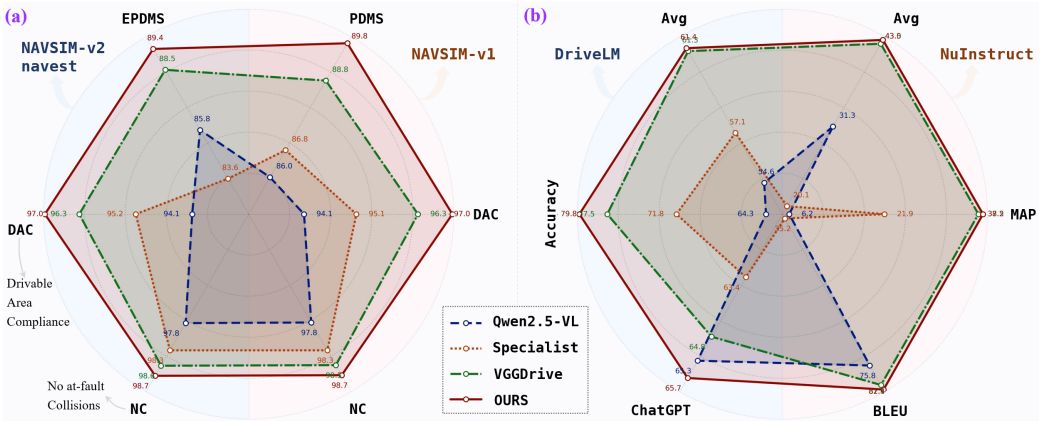


Figure 2: Quantitative comparison of LECDrive with state-of-the-art methods across three autonomous driving benchmarks, covering closed-loop planning, perception and prediction tasks.

in the latent space, where earlier stages provide increasingly rich context for subsequent stages, while later-stage supervision further refines earlier representations. **From a cognitive perspective, LECDrive establishes a latent cognitive trilogy within the VLM. Before generating driving actions, the model first learns to perceive the world like a visual feature extractor, understand it like a semantic segmenter, and finally measure it like a depth sensor.**

For trajectory prediction, we further design a hybrid formulation that combines discrete motion primitive retrieval with continuous offset refinement while preserving dense world understanding, thereby improving both the robustness and precision of trajectory planning. Extensive experiments demonstrate that LECDrive consistently and significantly improves baseline VLM performance across multiple autonomous driving benchmarks, including open-loop and closed-loop trajectory planning (Figure 2(a)), as well as language-driven perception and prediction tasks (Figure 2(b)). We summarize the contribution of our work as follows:

• We propose a new driving paradigm, Think Densely, Act Sparsely, which constructs dense world understanding in the latent space before generating sparse actions, providing a principled solution to dense information loss in VLA frameworks.

• We introduce an efficient and interpretable Latent Expert Cognitive Chain that injects a compact set of task-specific tokens to progressively internalize visual, semantic, and geometric knowledge, enabling dense world modeling in the latent space.

• Extensive experiments demonstrate consistent and significant performance gains of LECDrive across multiple autonomous driving benchmarks, including open-loop and closed-loop trajectory planning and language-driven perception and prediction tasks.

2 Related Works

2.1 Decoupled Action Modeling in VLA Driving

Vision-Language Models (VLMs) [8–10], with their rich world knowledge and strong reasoning capabilities, have significantly improved the generalization and interpretability of autonomous driving systems [11–13]. From early use as high-level semantic understanding modules to end-to-end Vision-Language-Action (VLA) frameworks that autoregressively generate driving trajectories [14–16], the loss of dense visual information caused by sparse supervision remains a persistent bottleneck. This limitation has motivated some approaches to introduce dedicated action decoders for trajectory prediction [15, 16, 22, 23]. ORION [22] addresses trajectory prediction by introducing a VAE-based generative action decoder to align the reasoning and action spaces. ReCogDrive [15] further incorporates a diffusion-based planner, augmented with VLM-driven driving priors, as the action decoder to mitigate the mismatch between language and action spaces, enabling the generation of smooth, stable, and safe trajectories. While improving trajectory prediction accuracy, these approaches decouple scene understanding and decision making into loosely coupled stages, limiting the transfer of VLM reasoning capabilities to trajectory planning.

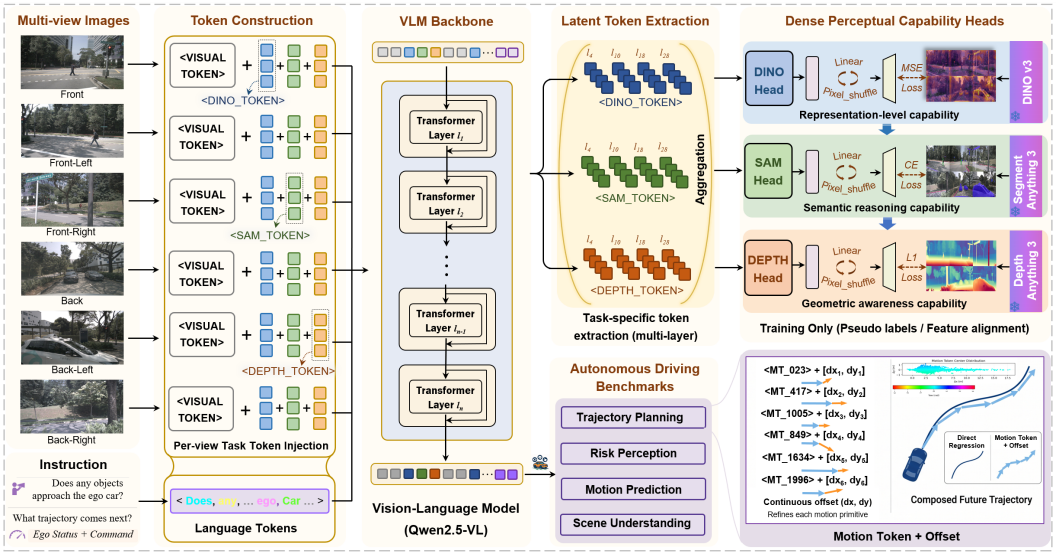


Figure 3: **Overview of LECDrive**. Three frozen dense expert models first generate task-specific pseudo-labels. A compact set of task-specific latent tokens is then injected into the visual tokens of each view. Task-specific latent embeddings from four VLM layers are aggregated and fed into dedicated perception heads for dense world prediction. For trajectory planning, LECDrive combines discrete motion primitives with continuous offset refinement to improve robustness and precision.

2.2 Explicit Geometry Augmentation for VLA Driving

To compensate for the limitations of VLMs in spatial geometric modeling, another line of work [4, 24–26] introduces explicit geometry augmentation, leveraging 3D information or auxiliary geometric supervision to enhance the understanding of scene structure. These approaches typically incorporate multimodal signals such as point clouds, occupancy grids, or depth maps to complement or constrain visual features, thereby improving spatial perception. VGGDrive [25] enhances cross-view geometric reasoning in VLMs by injecting features from 3D expert models [30] into the VLM architecture, thereby improving performance on multiple autonomous driving evaluation tasks. DrivePI [24] augments the visual representations of VLMs with point cloud and occupancy flow prediction to strengthen geometric perception, leading to improved open-loop performance on the nuScenes benchmark. However, such methods face two key challenges. First, explicitly integrating 3D information often incurs additional computational overhead, reducing inference efficiency [25]. Second, naively combining multiple geometry-related objectives can lead to task interference, making it difficult to learn a unified and stable representation space [24, 26]. More importantly, these approaches primarily focus on geometric enhancement and fail to address the fundamental issue of dense visual information loss when mapping from dense perception to sparse action representations.

3 Methodology

To enable efficient dense world modeling in autonomous driving, we propose LECDrive, a latent expert cognitive chain framework that progressively internalizes visual, semantic, and geometric knowledge through a compact set of task-specific latent tokens with minimal inference overhead. As illustrated in Figure 3, LECDrive consists of three components: 1) the VLM backbone (Qwen2.5-VL), 2) the Latent Expert Cognitive Chain, and 3) Dense Perception Heads. Details are provided below.

3.1 VLM Backbone

We adopt the Qwen2.5-VL-7B model [9] as the visual-language backbone for LECDrive. Qwen2.5-VL is a series of powerful multimodal large language models, it not only demonstrates exceptional visual understanding capabilities but also benefits from its open-source nature, which facilitates fine-tuning for task-specific adaptations. In our framework, the VLM receives multi-view driving images $I = \{I_c\}_{c=1}^{C}$ together with task-specific language instructions L as input. Specifically, for benchmarks built upon the nuScenes dataset, including *NuInstruct* [31], *DriveLM* [32], *OmniDrive* [23], and *nuScenes-Plan* [33], we adopt six surround-view images from the current frame ($C = 6$).

For trajectory planning tasks on NAVSIM [34], we instead use three front-view perspectives, namely front-left, front, and front-right ($C = 3$). In addition, for trajectory planning benchmarks such as *nuScenes-Plan* and NAVSIM, ego-vehicle states and high-level navigation commands are further incorporated into the language instructions. During instruction construction, multiple groups of compact task-specific latent tokens are inserted after the visual tokens corresponding to each view. The resulting tokens are subsequently fed into the base LLM to autoregressively generate the corresponding textual outputs. The model is optimized using the standard cross-entropy loss:

$$\mathcal{L}_{\text{CE}} = -\sum_{t=1}^{T} \log p_\theta(y_t | y_{<t},\ \{I_c\}_{c=1}^{C},\ L), \tag{1}$$

where y_t is the t-th output token, and p_θ is the probability predicted by the model given all previous tokens and the multimodal context (i.e., images and instructions).

3.2 Latent Expert Cognitive Chain

To enable VLMs to progressively internalize dense world knowledge with minimal additional inference cost, we propose the Latent Expert Cognitive Chain in LECDrive. The core idea is to inject a compact set of task-specific latent tokens into the VLM, forming a progressive cognitive chain that sequentially internalizes visual representation (DINOv3 [27]), semantic structure (SAM3 [28]), and spatial geometry knowledge (DepthAnything3 [29]) from foundational vision experts. Inspired by the hierarchical nature of human perception and curriculum learning, these expert stages are not optimized as independent objectives, but organized as an interdependent latent reasoning process, where higher-level supervision progressively refines lower-level representations.

Specifically, we first inject three groups of compact task-specific latent tokens after the visual tokens of each view, each containing n tokens dedicated to dense world modeling. These include *<DINO_TOKEN>* (F_t) for internalizing visual representation knowledge, *<SAM_TOKEN>* (S_t) for learning traffic-relevant semantic structures, and *<DEPTH_TOKEN>* (D_t) for capturing spatial geometric information. After token embedding, the corresponding latent embeddings ($\{F_e, S_e, D_e\} \in \mathbb{R}^{B\times C\cdot n\times D}$) are fed into the stacked Transformer layers of the VLM, producing hierarchical hidden states $\{H_i\}_{i=1}^{28}$. Subsequently, we explicitly extract task-specific embeddings from four Transformer layers ranging from shallow to deep and aggregate them across layers. This design is motivated by the hierarchical abstraction property naturally formed within VLM Transformers, where representations progressively evolve from local textures to global semantics. Shallow layers preserve rich fine-grained appearance details, intermediate layers capture part-level semantics and object structures, deeper layers become increasingly sensitive to object categories and spatial relationships, while the deepest layers encode high-level cross-modal reasoning capabilities. Therefore, relying on embeddings from a single layer would inevitably lead to systematic information loss under the extremely limited token budget. To obtain task-specific representations, latent embeddings ($\{F_i, S_i, D_i\} \in \mathbb{R}^{B\times C\cdot n\times D}$) are extracted from the hidden states $\{H_i\}_{i=1}^{28}$ using fixed positional ID masks corresponding to different expert tokens. Taking depth embeddings as an example.

$$D_i = \begin{cases} H_i, & M_{id}^{depth} = 1, \\ 0, & \text{otherwise}, \end{cases} \quad i \in \{4, 10, 18, 28\}, \tag{2}$$

where M_{id}^{depth} denotes a binary mask specific to the depth task, whose values are set to 1 at positions corresponding to the *<DEPTH_TOKEN>* and 0 otherwise. Subsequently, task-specific embeddings extracted from four Transformer layers are aggregated across layers ($\{F, S, D\} \in \mathbb{R}^{B\times C\cdot n\times 4\cdot D}$) and passed to the corresponding dense world modeling heads for dense prediction.

3.3 Dense Perception Capability Heads

To enable the injected task-specific latent tokens to fully internalize dense world knowledge, we design three lightweight dense perception heads for corresponding dense prediction tasks. Meanwhile, we leverage three state-of-the-art vision expert models to generate pseudo-labels, covering visual representation features F^{gt}, traffic-aware semantic segmentation S^{gt}, and metric depth map D^{gt}.

Representation Perception Head. We design a lightweight representation head to align visual embeddings $F \in \mathbb{R}^{B\times C\cdot n\times 4\cdot D}$ with the DINO feature F^{gt}. The input embeddings are first reshaped into a compact 2D feature map $F^{2d} \in \mathbb{R}^{B\cdot C\times h\times w\times 4\cdot D}$, where $n = h \times w$ and $2h = w$, followed by

Table 1: Performance comparison on the NAVSIM-v1 benchmark [34] under closed-loop trajectory evaluation, covering both E2E methods (gray) and supervised fine-tuned VLA models (green).

NAVSIM-v1	Input	NC↑	DAC↑	EP↑	TTC↑	Comf.↑	PDMS↑
PARA-Drive [35]	Camera	97.9	92.4	79.3	93.0	99.8	84.0
DRAMA [36]	Camera & Lidar	98.19	95.18	81.33	94.17	**100.0**	86.87
Hydra-MDP [37]	Camera & Lidar	98.3	96.0	78.7	94.6	**100.0**	86.5
DiffusionDrive [38]	Camera & Lidar	98.2	96.2	82.2	94.7	**100.0**	88.1
WoTE [7]	Camera & Lidar	98.5	96.8	81.9	94.9	99.9	88.3
Hydra-NeXt [37]	Camera & Lidar	98.1	97.7	81.8	94.6	**100.0**	88.6
Baseline [9]	Camera	97.83	94.08	81.0	94.04	99.98	86.04
ImagiDrive [39]	Camera	97.9	95.5	80.7	93.1	99.9	86.4
AutoVLA [19]	Camera	96.89	94.43	75.82	88.06	99.94	80.54
ReCogDrive [15]	Camera	98.3	95.1	81.1	94.3	**100.0**	86.8
AdaThinkDrive [16]	Camera	**98.9**	95.3	80.6	96.0	**100.0**	87.5
DriveVLA-W0 [40]	Camera	98.7	96.2	82.2	95.5	**100.0**	88.4
LatentVLA [41]	Camera & Lidar	98.2	95.9	79.4	94.8	99.9	86.6
OneDrive [42]	Camera	98.4	95.2	81.1	94.9	**100.0**	86.8
LaST-VLA [26]	Camera	98.7	95.4	80.5	95.6	**100.0**	86.8
VGGDrive [25]	Camera	98.55	96.3	82.92	95.59	99.98	88.76
OURS	Camera	98.72	**96.97**	**83.68**	**96.39**	99.99	**89.81**

Table 2: The performance comparison on the NAVSIM-v2 *navtest* [34] benchmark considers both SOTA end-to-end approaches and existing VLA models under supervised fine-tuning.

NAVSIM-v2	NC↑	DAC↑	DCC↑	TLC↑	EP↑	TTC↑	LK.↑	HC↑	EC↑	EPDMS↑
Ego-MLP [43]	93.1	77.9	92.7	99.6	86.0	91.5	89.4	98.3	85.4	64.0
TransFuser	96.9	89.9	97.8	99.7	87.1	95.4	92.7	98.3	87.2	76.7
Hydra-MDP++	97.2	**97.5**	99.4	99.6	83.1	96.5	94.4	98.2	70.9	81.4
DriveSuprim [44]	97.5	96.5	99.4	99.6	**88.4**	96.6	95.5	98.3	77.0	83.1
ReCogDrive	98.3	95.2	**99.5**	99.8	87.1	97.5	96.6	98.3	86.5	83.6
DiffusionDrive	98.2	96.3	99.4	99.8	87.4	97.4	97.0	98.3	**87.7**	88.3
VGGDrive	98.56	96.30	99.38	99.85	87.85	98.18	97.29	98.31	83.40	88.50
OURS	**98.73**	96.97	**99.49**	**99.88**	87.89	**98.43**	**97.34**	**98.34**	84.50	**89.44**

a two-stage upsampling pipeline. In each stage, Pixel Shuffle is employed for spatial upsampling, after which the spatial tokens are linearly projected to enhance local representation capacity.

$$X' = \phi(PS(F^{2d})), \tag{3}$$

where $PS(\cdot)$ denotes the Pixel Shuffle operation for spatial upsampling and $\phi(\cdot)$ denotes token flattening followed by a linear projection. The resulting high-resolution features F^{pre} are finally projected into the DINO feature space F^{gt} for representation alignment. To encourage the hidden representations to preserve semantic consistency with DINO features, we minimize the mean squared error (MSE) between the predicted features and the target DINO representations:

$$\mathcal{L}_{\text{DINO}} = \frac{1}{N} \left\| F^{pre} - F^{gt} \right\|_2^2. \tag{4}$$

Semantic Perception Head. We design a lightweight semantic perception head to map compact task-specific embeddings $S \in \mathbb{R}^{B\times C\cdot n\times 4\cdot D}$ into traffic-aware semantic segmentation predictions S^{gt} via stacked Eq. 3 blocks. In this task, we focus on six semantic categories that are closely related to autonomous driving, including *Vehicles*, *Pedestrians*, *Traffic Signs*, *Traffic Lights*, *Road Markings*, and *Road Edges*. We optimize the semantic segmentation branch using a weighted cross-entropy loss to alleviate the class imbalance issue. Specifically, lower weight is assigned to the background category, while foreground classes share identical weights.

$$\mathcal{L}_{\text{SAM}} = -\frac{1}{N} \sum_{i=1}^{N} w_{y_i} \log p_{i,y_i}, \tag{5}$$

where N denotes the number of pixels, y_i is the ground-truth label of pixel i, p_{i,y_i} denotes the predicted probability for the corresponding class, and w_{y_i} is the class-specific weighting factor. We set the background class weight to 0.2 and assign a weight of 1.0 to all remaining classes.

Table 3: Comparison with sota methods on the NuInstruct dataset [31] for risk object perception and state prediction. * indicates $\max((\text{Accuracy} + \text{MAP} + \text{BLEU} - \text{MAE})/4, 0)$.

Dataset	Metrics	Zero-shot Models			Fine-tuned Models			
		GPT-4o [45]	RoboTron [46]	Qwen2.5-VL	Baseline	InMLLM [31]	VGGDrive	OURS
NuInstruct	**MAP↑**	0	0	0	6.15	21.93	37.49	**38.25**
	Accuracy↑	10.64	2.57	0.63	47.71	32.48	56.37	**57.02**
	MAE↓	9.93	19.36	24.10	4.35	9.08	**3.08**	3.57
	BLEU↑	7.08	8.06	5.56	75.75	35.20	81.13	**82.38**
	Average*↑	1.95	0	0	31.32	20.13	42.98	**43.52**

Table 4: The performance comparison on the DriveLM dataset [32] is conducted against existing state-of-the-art (SOTA) methods to evaluate risk object perception, action prediction, and planning.

Dataset	Metrics	Fine-tuned Models							
		Baseline	DriveLM	TMLMM [47]	OminiD [23]	FSDrive [48]	IVL4Drive [49]	VGGDrive	OURS
DriveLM	Accuracy↑	64.35	52.30	59.60	70.00	71.77	73.39	77.50	**79.81**
	ChatGPT↑	65.30	55.35	58.44	65.00	63.42	65.25	64.76	**65.69**
	Language↑	44.90	45.13	46.38	-	52.77	48.56	**56.69**	56.53
	Match↑	34.54	35.73	35.73	37.00	39.19	47.65	**49.77**	47.73
	Average↑	54.59	47.20	51.50	56.00	57.12	60.02	61.26	**61.41**

Geometric Perception Head. Since DepthAnything3 [29] supports camera-parameter-conditioned depth estimation, and such information is naturally available in autonomous driving scenarios, camera parameters are incorporated when generating depth pseudo-labels. Meanwhile, a lightweight depth estimation head is designed to map depth-specific embeddings into metric depth predictions, implemented using stacked Eq. 3 blocks. We supervise the depth estimation branch using a masked Smooth L_1 loss. Only valid depth regions are involved in optimization, where depth values are constrained within a predefined range.

$$\mathcal{L}_{\text{DEPTH}} = \frac{\sum_{i=1}^{N} M_i \cdot \text{SmoothL1}\left(D_i^{pre}, D_i^{gt}\right)}{\sum_{i=1}^{N} M_i + \epsilon}, \tag{6}$$

where D_i^{pre} and D_i^{gt} denote the predicted and ground-truth depth values at pixel i, respectively, M_i is the validity mask, and ϵ is a small constant for numerical stability. We set $M_i = 1$ when $0 < D_i^{gt} \leq 60$, and $M_i = 0$ otherwise. The overall training objective is defined as:

$$\mathcal{L} = \lambda_1 \cdot \mathcal{L}_{\text{CE}} + \lambda_2 \cdot \mathcal{L}_{\text{DINO}} + \lambda_3 \cdot \mathcal{L}_{\text{SAM}} + \lambda_4 \cdot \mathcal{L}_{\text{DEPTH}}, \tag{7}$$

where $\{\lambda_i\}_{i=1}^{4}$ represents the weighting factor for each objective in the joint training process.

4 Experiments

4.1 Implementation Details

In our experiments, we adopt Qwen2.5-VL-7B [9] as the base Vision-Language Model. The proposed latent expert cognitive chain is instantiated with three progressive expert modules: a feature representation expert based on DINOv3 [27], a semantic expert based on SAM 3 [28], and a geometric expert based on DepthAnything 3 [29]. Specifically, DINOv3 employs a ViT-based model with 840M parameters pretrained on the large-scale web dataset LVD-1689M. For geometric modeling, we adopt the DepthAnything3 (DA3NESTED-GIANT-LARGE) variant with 1.4B parameters. During training, the learning rate is set to 5×10^{-5} and the model is trained for 5 epochs. The number of task-specific latent tokens for each expert n is set to 8. The loss weights for different components are set to $\lambda_1 = 1$, $\lambda_2 = 0.5$, $\lambda_3 = 0.5$, and $\lambda_4 = 1.5$. All experiments are conducted on 8 NVIDIA H200 GPUs. For trajectory planning, we cluster trajectory data into 2048 motion primitives, transforming direct waypoint prediction into a hybrid formulation of discrete motion primitive retrieval with continuous offset refinement. Details are provided in the Appendix A.3.

4.2 Dataset and Metric

To validate the effectiveness of the proposed Think Densely, Act Sparsely paradigm—constructing dense world understanding before generating sparse driving actions—and the latent expert cognitive

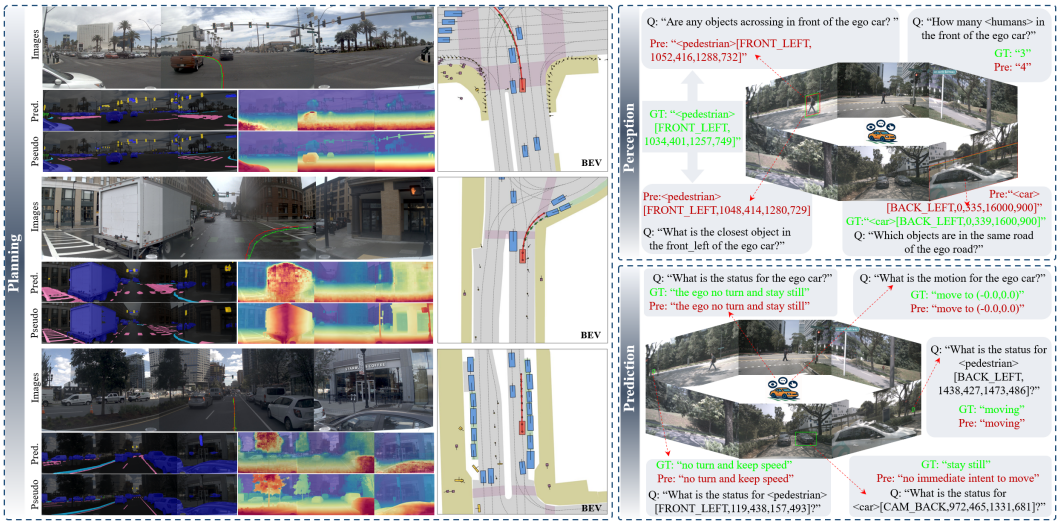


Figure 4: Visualization of LECDrive across various driving tasks. Besides trajectory planning, we showcase the model's internal dense world modeling results, including semantic segmentation and depth estimation generated by dedicated heads, along with risk object perception and state prediction.

chain architecture, we conduct experiments across various autonomous driving tasks on five mainstream benchmarks. These benchmarks include language evaluation benchmarks: NuInstruct [31], DriveLM [32], and Omnidrive [23], as well as two trajectory planning tasks: NuScenes-Plan [33] and NAVSIM [34]. The NuInstruct and DriveLM datasets contain numerous cases for evaluating risk object perception, state and motion prediction (e.g., Figure 4). The Omnidrive dataset primarily focuses on scene description tasks related to captioning in driving scenarios. The NuScenes-Plan and NAVSIM datasets focus on open-loop and closed-loop trajectory planning tasks, respectively. To ensure a fair comparison, we evaluate LECDrive on five mainstream datasets using their standard metrics. Detailed dataset and metric descriptions are provided in Appendix A.1.

4.3 Performance Comparison

NAVSIM. As reported in Table 1, the closed-loop trajectory comparison is shown. 1) Compared to AdaThinkDrive, a representative decoupled approach that relies on an additional action decoder for trajectory prediction, LECDrive achieves a **+2.31** improvement in PDMS. 2) Compared with VGGDrive, which enhances spatial perception by injecting 3D features into VLMs, LECDrive still achieves a **+1.05** gain in PDMS. Under the more stringent NAVSIM v2 *navtest* benchmark (Table 2), LECDrive further outperforms VGGDrive by **+0.94** in EPDMS. See appendix A.2 for more details.

NuInstruct. As shown in Table 3, the base VLM performs poorly on tasks such as risk object perception (MAP) and state prediction, and exhibits only marginal gains even after fine-tuning on task-specific datasets. In contrast, the proposed LECDrive achieves substantial improvements on the key MAP and Accuracy metrics, with gains of **+32.10** and **+9.31**, respectively.

DriveLM. This benchmark is crucial for evaluating a model's capability in object perception, action prediction and planning. As reported in Table 4, LECDrive substantially surpasses the baseline, improving the Accuracy and Average metrics by **15.46** and **6.82**, respectively.

OmniDrive and NuScenes-plan. These two benchmarks primarily serve as complementary evaluations in our experiments. Detailed descriptions and comparisons are provided in Appendix A.2.

LECDrive's consistent advantages across multiple benchmarks suggest that a progressive latent expert cognitive chain, which enhances dense world modeling in VLMs without incurring additional inference cost, is both effective and promising for VLA-based autonomous driving.

Visualization Analysis. Figure 4 visualizes the performance of LECDrive across multiple autonomous driving tasks. For trajectory planning, we further present the model's dense world modeling results, including semantic segmentation and depth estimation alongside expert pseudo-labels. With only 8 task-specific latent tokens, LECDrive effectively enables dense world understanding in VLMs. Moreover, the model also demonstrates strong performance on risk object perception and state prediction tasks. Details are provided in the Appendix A.2

Table 5: Ablation study on the effectiveness of the proposed key components for closed-loop planning.

ID	Baseline	MT	Offset	DINO v3	SAM 3	DA 3	NC↑	DAC↑	EP↑	TTC↑	Comf.↑	PDMS↑
1	✓						97.83	94.08	81.00	94.04	99.98	86.04
2	✓	✓					98.44	94.95	81.45	95.72	99.99	87.44
3	✓	✓	✓				98.62	96.06	82.38	95.78	**100.00**	88.49
4	✓	✓	✓	✓			98.53	96.57	82.99	95.97	99.99	89.07
5	✓	✓	✓	✓	✓		98.68	96.76	83.27	95.97	99.98	89.31
6	✓	✓	✓		✓	✓	98.65	96.85	83.54	96.08	99.99	89.53
7	✓	✓	✓	✓		✓	98.69	96.80	83.48	96.29	**100.00**	89.57
8	✓	✓	✓	✓	✓	✓	**98.72**	**96.97**	**83.68**	**96.39**	99.99	**89.81**

Table 6: Ablation Study on Architectural Variants and Key Hyperparameters in LECDrive.

ID	Methods	NC↑	DAC↑	EP↑	TTC↑	Comf.↑	PDMS↑
1	Baseline	97.83	94.08	81.00	94.04	99.98	86.04
2	Visual tokens + Task heads	98.58	96.28	82.73	95.82	99.99	88.73
3	Task tokens (l_{28}) + Task heads	98.59	96.78	83.14	95.94	99.99	89.26
4	Task tokens = 2	98.57	96.54	82.93	96.16	99.99	89.13
5	Qwen2.5-VL-3B	98.66	96.66	83.19	95.99	**100.00**	89.21
6	OURS	**98.72**	**96.97**	**83.68**	**96.39**	99.99	**89.81**

4.4 Ablation Study

We perform extensive ablation studies on the NAVSIM-v1 [34] closed-loop benchmark to systematically evaluate the effectiveness of the proposed architecture and its core components.

Effectiveness of the Latent Expert Cognitive Chain. Table 5 presents the ablation study on the effectiveness of the three chained expert modules in LECDrive for closed-loop trajectory planning. Specifically, comparing ID-3 and ID-4 shows that introducing the dense visual representation expert, DINOv3, yields a **+0.58** improvement in PDMS. Building upon this, incorporating the semantic structure expert SAM 3 further improves performance by **+0.24**. When the geometric expert DepthAnything 3 is additionally integrated, the model achieves another significant gain of **+0.50**, reaching a state-of-the-art PDMS of **89.81**. Comparing ID-6 and ID-7 indicates that visual representation and semantic experts exhibit shared characteristics when internalized by VLMs, yet removing either expert leads to performance degradation, highlighting their complementary contributions. These results demonstrate that progressively introducing dense expert modules in a chained manner—mirroring the hierarchical nature of human perception—enables VLMs to incrementally internalize dense world knowledge, leading to consistent performance improvements.

Effectiveness of the Hybrid Trajectory Formulation. Table 5 also presents an ablation study on the proposed hybrid trajectory formulation. Replacing direct coordinate regression with discrete motion primitive retrieval (ID-2) improves PDMS by **+1.40**, demonstrating the benefit of structured discrete trajectory modeling, although this formulation alone remains less competitive with sota methods (Table 1). Further introducing continuous offset refinement boosts PDMS to **88.49**, making the model competitive with strong expert-designed methods (88.49 *vs.* 88.76 for VGGDrive [25]). These results suggest that discrete primitives provide robustness, while continuous offsets recover fine-grained precision. Overall, the hybrid formulation effectively balances robustness and accuracy, significantly improving trajectory prediction performance. Additional details are provided in Appendix A.3.

Effectiveness of the Architectural Variants and Key Hyperparameters. Table 6 presents ablation studies on architectural variants and key hyperparameters of LECDrive. Removing task-specific latent tokens (ID-2) causes a significant performance drop (**-1.08** PDMS), indicating severe task interference without explicit latent specialization. Using only final-layer latent tokens without multi-layer aggregation (ID-3) further reduces performance by **-0.55** PDMS, demonstrating the importance of integrating shallow and deep VLM features. ID-4 and ID-5 further analyze the effects of latent token quantity and base model scale, both of which influence overall performance.

Disentangling Expert Knowledge from Trajectory Modeling. To isolate the contribution of latent expert knowledge internalization from the hybrid trajectory formulation, we evaluate the expert modules without MT+Offset. As shown in Table 7(a), DINOv3, SAM 3, and DepthAnything 3 individually improve PDMS from 86.04 to 87.98, 88.17, and 88.47, respectively. Combining all three

Table 7: Further ablations on expert knowledge internalization and causal ordering on NAVSIM-v1.

Configuration	NC↑	DAC↑	EP↑	TTC↑	PDMS↑
(a) Expert supervision without MT+Offset					
Baseline	97.83	94.08	81.00	94.04	86.04
+ DINOv3	98.38	95.49	82.71	95.01	87.98
+ SAM 3	98.21	95.80	83.03	97.85	88.17
+ DepthAnything 3	98.60	96.01	82.39	95.81	88.47
+ All Experts	98.59	96.51	82.97	96.05	**89.15**
(b) Expert ordering with MT+Offset					
DINO → DEPTH → SAM	98.58	96.54	82.93	96.26	89.18
SAM → DINO → DEPTH	98.63	96.93	83.59	96.20	89.66
DINO → SAM → DEPTH	**98.72**	**96.97**	**83.68**	**96.39**	**89.81**

experts further achieves 89.15 PDMS, yielding a substantial gain of +3.11 over the baseline. These results demonstrate that dense expert knowledge internalization improves planning performance independently of the hybrid trajectory representation.

Effect of Expert Ordering. We further investigate the importance of the proposed causal expert ordering. As shown in Table 7(b), changing the representation–semantics–geometry order to representation–geometry–semantics or semantics–representation–geometry decreases PDMS from 89.81 to 89.18 and 89.66, respectively. These results indicate that the ordering of expert-specific latent tokens influences knowledge integration, supporting the effectiveness of the proposed progressive cognitive hierarchy.

Computational Efficiency. Without MT+Offset, LECDrive increases training time from 2.6 to 5.4 hours per epoch, primarily due to online dense pseudo-label generation using three frozen vision foundation models, rather than the lightweight perception heads. This overhead can be substantially reduced by precomputing and caching expert pseudo-labels. Importantly, all expert models and dense prediction heads are discarded during inference, leaving only compact expert-specific latent tokens that account for 2.7% of the total input tokens. Consequently, inference time increases only marginally from 0.80 to 0.85 seconds, demonstrating that LECDrive retains internalized dense world knowledge without substantial inference overhead.

5 Limitations

Although LECDrive effectively alleviates the mismatch between dense perception and sparse supervision under the **Think Densely, Act Sparsely** paradigm, several limitations remain. 1) The method relies on high-quality pseudo-labels and external expert models, whose capacity and generalization inherently bound performance. 2) A limited number of latent tokens may introduce representation bottlenecks (Figure 4), while the scaling behavior of token quantity is not explored due to computational constraints. 3) The current design focuses on single-frame or weak temporal modeling, leaving dynamic consistency and unified multi-view scene understanding underexplored. Future work will explore more expressive latent representations, adaptive token allocation, and unified spatio-temporal modeling to improve robustness and generalization in complex real-world scenarios.

6 Conclusion

In this paper, we revisit the fundamental limitation of VLA models for autonomous driving, namely the mismatch between dense perception and sparse supervision, which leads to the loss of critical scene information. To address this issue, we propose a new paradigm, Think Densely, Act Sparsely, which emphasizes constructing dense world representations in the latent space prior to action generation. Building upon this paradigm, we introduce LECDrive, an efficient and interpretable latent expert cognitive chain framework that progressively internalizes visual, semantic, and geometric knowledge via a compact set of task-specific tokens. Extensive experiments across multiple autonomous driving benchmarks demonstrate that LECDrive consistently improves performance on trajectory planning, perception, and prediction tasks. More broadly, our findings suggest that explicitly modeling structured dense representations in the latent space is a promising direction for advancing VLA-based autonomous driving. We hope this work inspires future research toward unified and efficient frameworks that bridge perception and action through richer internal world modeling.

Acknowledgments and Disclosure of Funding

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 62376186.

The authors declare no competing interests.

References

[1] K. Chitta, A. Prakash, B. Jaeger, Z. Yu, K. Renz, and A. Geiger, "Transfuser: Imitation with transformer-based sensor fusion for autonomous driving," *IEEE TPAMI*, vol. 45, no. 11, pp. 12 878–12 895, 2022.

[2] C. Min, D. Zhao, L. Xiao, J. Zhao, X. Xu, Z. Zhu, L. Jin, J. Li, Y. Guo, J. Xing *et al.*, "Driveworld: 4d pre-trained scene understanding via world models for autonomous driving," in *CVPR*, 2024, pp. 15 522–15 533.

[3] Y. Wang, J. He, L. Fan, H. Li, Y. Chen, and Z. Zhang, "Driving into the future: Multiview visual forecasting and planning with world model for autonomous driving," in *CVPR*, 2024, pp. 14 749–14 759.

[4] C. Dang, J. Wang, G. Li, Z. Hou, Z. You, H. Ye, J. Ma, L. Chen, and Y. Wang, "Sparseocc-cvla: Bridging occupancy and vision-language models via sparse queries for unified 4d scene understanding and planning," *arXiv preprint arXiv:2601.06474*, 2026.

[5] X. Zhou, D. Liang, S. Tu, X. Chen, Y. Ding, D. Zhang, F. Tan, H. Zhao, and X. Bai, "Hermes: A unified self-driving world model for simultaneous 3d scene understanding and generation," in *ICCV*, 2025, pp. 27 817–27 827.

[6] C. Shi, S. Shi, K. Sheng, B. Zhang, and L. Jiang, "Drivex: Omni scene modeling for learning generalizable world knowledge in autonomous driving," in *CVPR*, 2025, pp. 28 599–28 609.

[7] Y. Li, Y. Wang, Y. Liu, J. He, L. Fan, and Z. Zhang, "End-to-end driving with online trajectory evaluation via bev world model," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025, pp. 27 137–27 146.

[8] J. Zhu, W. Wang, Z. Chen, Z. Liu, S. Ye, L. Gu, H. Tian, Y. Duan, W. Su, J. Shao *et al.*, "Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models," *arXiv preprint arXiv:2504.10479*, 2025.

[9] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang *et al.*, "Qwen2. 5-vl technical report," *arXiv preprint arXiv:2502.13923*, 2025.

[10] P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge *et al.*, "Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution," *arXiv preprint arXiv:2409.12191*, 2024.

[11] B. Jiang, S. Chen, B. Liao, X. Zhang, W. Yin, Q. Zhang, C. Huang, W. Liu, and X. Wang, "Senna: Bridging large vision-language models and end-to-end autonomous driving," *arXiv preprint arXiv:2410.22313*, 2024.

[12] B. Jiang, S. Chen, Q. Zhang, W. Liu, and X. Wang, "Alphadrive: Unleashing the power of vlms in autonomous driving via reinforcement learning and reasoning," *arXiv preprint arXiv:2503.07608*, 2025.

[13] A.-M. Marcu, L. Chen, J. Hünermann, A. Karnsund, B. Hanotte, P. Chidananda, S. Nair, V. Badrinarayanan, A. Kendall, J. Shotton *et al.*, "Lingoqa: Visual question answering for autonomous driving," in *ECCV*. Springer, 2024, pp. 252–269.

[14] K. Renz, L. Chen, E. Arani, and O. Sinavski, "Simlingo: Vision-only closed-loop autonomous driving with language-action alignment," in *CVPR*, 2025, pp. 11 993–12 003.

[15] Y. Li, K. Xiong, X. Guo, F. Li, S. Yan, G. Xu, L. Zhou, L. Chen, H. Sun, B. Wang *et al.*, "Recogdrive: A reinforced cognitive framework for end-to-end autonomous driving," in *ICLR*, 2026.

[16] Y. Luo, F. Li, S. Xu, Z. Lai, L. Yang, Q. Chen, Z. Luo, Z. Xie, S. Jiang, J. Liu *et al.*, "Adathinkdrive: Adaptive thinking via reinforcement learning for autonomous driving," *arXiv preprint arXiv:2509.13769*, 2025.

[17] J.-J. Hwang, R. Xu, H. Lin, W.-C. Hung, J. Ji, K. Choi, D. Huang, T. He, P. Covington, B. Sapp *et al.*, "Emma: End-to-end multimodal model for autonomous driving," *arXiv preprint arXiv:2410.23262*, 2024.

[18] Y. Ma, Y. Cao, J. Sun, M. Pavone, and C. Xiao, "Dolphins: Multimodal language model for driving," in *ECCV*. Springer, 2024, pp. 403–420.

[19] Z. Zhou, T. Cai, S. Z. Zhao, Y. Zhang, Z. Huang, B. Zhou, and J. Ma, "Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning," in *NeurIPS*, 2025.

[20] K. Renz, L. Chen, A.-M. Marcu, J. Hünermann, B. Hanotte, A. Karnsund, J. Shotton, E. Arani, and O. Sinavski, "Carllava: Vision language models for camera-only closed-loop driving," *arXiv preprint arXiv:2406.10165*, 2024.

[21] C. Dang, S. Ang, Y. Li, H. Tian, J. Wang, G. Li, H. Ye, J. Ma, L. Chen, and Y. Wang, "Drivefine: Refining-augmented masked diffusion vla for precise and robust driving," in *ECCV*. Springer, 2026.

[22] H. Fu, D. Zhang, Z. Zhao, J. Cui, D. Liang, C. Zhang, D. Zhang, H. Xie, B. Wang, and X. Bai, "Orion: A holistic end-to-end autonomous driving framework by vision-language instructed action generation," in *ICCV*, 2025.

[23] S. Wang, Z. Yu, X. Jiang, S. Lan, M. Shi, N. Chang, J. Kautz, Y. Li, and J. M. Alvarez, "Omnidrive: A holistic vision-language dataset for autonomous driving with counterfactual reasoning," in *CVPR*, 2025, pp. 22 442–22 452.

[24] Z. Liu, R. Huang, R. Yang, S. Yan, Z. Wang, L. Hou, D. Lin, X. Bai, and H. Zhao, "Drivepi: Spatial-aware 4d mllm for unified autonomous driving understanding, perception, prediction and planning," *arXiv preprint arXiv:2512.12799*, 2025.

[25] J. Wang, G. Li, Z. Huang, C. Dang, H. Ye, Y. Han, and L. Chen, "Vggdrive: Empowering vision-language models with cross-view geometric grounding for autonomous driving," in *CVPR*, 2026.

[26] Y. Luo, F. Li, S. Xu, Y. Ji, Z. Zhang, B. Wang, Y. Shen, J. Cui, L. Chen, G. Chen *et al.*, "Last-vla: Thinking in latent spatio-temporal space for vision-language-action in autonomous driving," *arXiv preprint arXiv:2603.01928*, 2026.

[27] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa *et al.*, "Dinov3," *arXiv preprint arXiv:2508.10104*, 2025.

[28] N. Carion, L. Gustafson, Y.-T. Hu, S. Debnath, R. Hu, D. Suris, C. Ryali, K. V. Alwala, H. Khedr, A. Huang *et al.*, "Sam 3: Segment anything with concepts," *arXiv preprint arXiv:2511.16719*, 2025.

[29] H. Lin, S. Chen, J. Liew, D. Y. Chen, Z. Li, G. Shi, J. Feng, and B. Kang, "Depth anything 3: Recovering the visual space from any views," *arXiv preprint arXiv:2511.10647*, 2025.

[30] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny, "Vggt: Visual geometry grounded transformer," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 5294–5306.

[31] X. Ding, J. Han, H. Xu, X. Liang, W. Zhang, and X. Li, "Holistic autonomous driving understanding by bird's-eye-view injected multi-modal large models," in *CVPR*, 2024, pp. 13 668–13 677.

[32] C. Sima, K. Renz, K. Chitta, L. Chen, H. Zhang, C. Xie, J. Beißwenger, P. Luo, A. Geiger, and H. Li, "Drivelm: Driving with graph visual question answering," in *ECCV*. Springer, 2024, pp. 256–274.

[33] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *CVPR*, 2020, pp. 11 621–11 631.

[34] D. Dauner, M. Hallgarten, T. Li, X. Weng, Z. Huang, Z. Yang, H. Li, I. Gilitschenski, B. Ivanovic, M. Pavone *et al.*, "Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking," *NeurIPS*, vol. 37, pp. 28 706–28 719, 2024.

[35] X. Weng, B. Ivanovic, Y. Wang, Y. Wang, and M. Pavone, "Para-drive: Parallelized architecture for real-time autonomous driving," in *CVPR*, 2024, pp. 15 449–15 458.

[36] C. Yuan, Z. Zhang, J. Sun, S. Sun, Z. Huang, C. D. W. Lee, D. Li, Y. Han, A. Wong, K. P. Tee *et al.*, "Drama: An efficient end-to-end motion planner for autonomous driving with mamba," *arXiv preprint arXiv:2408.03601*, 2024.

[37] Z. Li, K. Li, S. Wang, S. Lan, Z. Yu, Y. Ji, Z. Li, Z. Zhu, J. Kautz, Z. Wu *et al.*, "Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation," *arXiv preprint arXiv:2406.06978*, 2024.

[38] B. Liao, S. Chen, H. Yin, B. Jiang, C. Wang, S. Yan, X. Zhang, X. Li, Y. Zhang, Q. Zhang *et al.*, "Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving," in *CVPR*, 2025, pp. 12 037–12 047.

[39] J. Li, B. Zhang, X. Jin, J. Deng, X. Zhu, and L. Zhang, "Imagidrive: A unified imagination-and-planning framework for autonomous driving," *arXiv preprint arXiv:2508.11428*, 2025.

[40] Y. Li, S. Shang, W. Liu, B. Zhan, H. Wang, Y. Wang, Y. Chen, X. Wang, Y. An, C. Tang *et al.*, "Drivevla-w0: World models amplify data scaling law in autonomous driving," in *ICLR*, 2026.

[41] C. Xie, B. Sun, T. Li, J. Wu, Z. Hao, X. Lang, and H. Li, "Latentvla: Efficient vision-language models for autonomous driving via latent action prediction," *arXiv preprint arXiv:2601.05611*, 2026.

[42] Y. Zhang, X. Chen, J. Gao, H. Wang, F. Ge, W. Hu, S. Shi, and Z. Zhang, "Onedrive: Unified multi-paradigm driving with vision-language-action models," *arXiv preprint arXiv:2604.17915*, 2026.

[43] Z. Li, Z. Yu, S. Lan, J. Li, J. Kautz, T. Lu, and J. M. Alvarez, "Is ego status all you need for open-loop end-to-end autonomous driving?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 14 864–14 873.

[44] W. Yao, Z. Li, S. Lan, Z. Wang, X. Sun, J. M. Alvarez, and Z. Wu, "Drivesuprim: Towards precise trajectory selection for end-to-end planning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 14, 2026, pp. 11 910–11 918.

[45] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford *et al.*, "Gpt-4o system card," *arXiv preprint arXiv:2410.21276*, 2024.

[46] Z. Huang, C. Feng, F. Yan, B. Xiao, Z. Jie, Y. Zhong, X. Liang, and L. Ma, "Robotron-drive: All-in-one large multimodal model for autonomous driving," in *ICCV*, 2025, pp. 8011–8021.

[47] A. Ishaq, J. Lahoud, F. S. Khan, S. Khan, H. Cholakkal, and R. M. Anwer, "Tracking meets large multimodal models for driving scenario understanding," *arXiv preprint arXiv:2503.14498*, 2025.

[48] S. Zeng, X. Chang, M. Xie, X. Liu, Y. Bai, Z. Pan, M. Xu, and X. Wei, "Futuresightdrive: Thinking visually with spatio-temporal cot for autonomous driving," in *NeurIPS*, 2025.

[49] J. Li, Z. Li, and T. Lu, "Driving with internvl: Oustanding champion in the track on driving with language of the autonomous grand challenge at cvpr 2024," *arXiv preprint arXiv:2412.07247*, 2024.

[50] Y. Hu, J. Yang, L. Chen, K. Li, C. Sima, X. Zhu, S. Chai, S. Du, T. Lin, W. Wang *et al.*, "Planning-oriented autonomous driving," in *CVPR*, 2023, pp. 17 853–17 862.

[51] B. Jiang, S. Chen, Q. Xu, B. Liao, J. Chen, H. Zhou, Q. Zhang, W. Liu, C. Huang, and X. Wang, "Vad: Vectorized scene representation for efficient autonomous driving," in *ICCV*, 2023, pp. 8340–8350.

A Appendix

To further complement the main content of this paper, we provide additional materials to elaborate on several key aspects of our method.Specifically, Sec. A.1 presents a more detailed description of the application of LECDrive across five mainstream autonomous driving benchmarks, along with their corresponding evaluation metrics. Sec. A.2 provides additional quantitative comparisons between LECDrive and representative methods on the nuScenes open-loop trajectory planning benchmark and the OmniDrive language evaluation benchmark. Sec. A.3 further introduces the hybrid formulation for trajectory planning, which combines discrete motion primitive retrieval with continuous offset refinement, along with supplementary visualizations.

A.1 Dataset and Metric

To assess the performance of LECDrive across various attributes within the autonomous driving domain, we conduct evaluations on five prominent benchmarks. These benchmarks cover task-specific scenarios, including scene understanding, risk object perception, action and state prediction, and trajectory planning.

NAVSIM-v1. This dataset [34] is a real-world, planning-focused dataset derived from OpenScene, which is a compact version of nuPlan, the largest publicly available annotated driving dataset (1,192 and 136 scenarios for training and testing). NAVSIM is designed to evaluate the performance of autonomous driving systems in complex, dynamic scenarios. It utilizes a combination of eight cameras providing a 360° field of view (FOV) and a merged LiDAR point cloud from five sensors. NAVSIM specifically targets challenging driving situations with dynamic driving intentions, while deliberately excluding simpler, static scenarios such as stationary scenes or constant-speed driving. The NAVSIM benchmark provides a nonreactive simulation environment and employs the Predictive Driver Model Score (PDMS) as its closed-loop planning metric:

$$PDMS = NC \times DAC \times \left(\frac{5\times EP+5\times TTC+2\times C}{12}\right) \tag{8}$$

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

NAVSIM-v2. In NAVSIM-v2 navtest, the Enhanced Planning and Driving Metric System (EPDMS) extends the original PDMS by introducing several new evaluation components, including two weighted metrics—Lane Keeping (LK) and Extended Comfort (EC)—two multiplicative metrics—Driving Direction Compliance (DDC) and Traffic Light Compliance (TLC)—as well as a false-positive penalty filtering mechanism. The Lane Keeping (LK) subscore penalizes prolonged deviations from the lane centerline. In addition, NAVSIM v2 places greater emphasis on driving comfort by extending the original Comfort (C) metric to History Comfort (HC), which evaluates the consistency between the planned trajectory and the vehicle's motion history. The newly introduced Extended Comfort (EC) further measures temporal consistency by comparing trajectory outputs and corresponding dynamic states across consecutive frames, where discrepancies in acceleration, jerk, and related dynamics lead to lower scores. Moreover, the Driving Direction Compliance (DDC) and Traffic Light Compliance (TLC) metrics expand the range of detectable and penalized driving violations. To reduce false-positive penalties, NAVSIM v2 disables penalties in cases where human drivers would also violate traffic rules. This design prevents unfair penalization in scenarios where rule violations are necessary to achieve valid driving objectives, such as temporarily entering an oncoming lane to bypass obstacles.

$$EPDMS = NC \times DAC \times DDC \times TLC \times \left(\frac{5\times EP+5\times TTC+2\times LK+2\times HC+2\times EC}{16}\right) \tag{9}$$

NuInstruct. This dataset [31] samples a total of 11,850 keyframes from 850 videos within the NuScenes dataset. It includes a wide range of challenging samples, such as cross-view risk perception, target distance estimation, agent and ego state prediction, target motion prediction, and reasoning tasks. The dataset effectively reflects the model's ability to perform cross-view understanding and analyze 3D geometric scene information. NuInstruct uses a variety of metrics to evaluate different tasks, including Mean Absolute Error (MAE) for regression tasks, accuracy for classification tasks, Mean Average Precision (mAP) for detection tasks, and a rule-based BLEU metric for captioning tasks.

Table 8: Performance comparison on nuScenes open-loop planning, with metrics from BEV-Planner's reproduced results [43].

nuScenes		L2 ↓				Collision (%) ↓				Intersection (%) ↓			
		1s	2s	3s	avg	1s	2s	3s	avg	1s	2s	3s	avg
UniAD [50]	E2E	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 [51]	E2E	0.17	0.34	0.60	0.37	0.04	0.27	0.67	0.33	0.21	2.13	5.06	2.47
BEV-Planner	E2E	0.16	0.32	0.57	0.35	0.00	0.29	0.73	0.34	0.35	2.62	6.51	3.16
Baseline	VLA	**0.14**	0.34	0.60	0.36	0.06	0.21	0.96	0.41	0.66	2.58	5.43	2.89
OmniDrive-Q	VLA	**0.14**	0.29	0.55	0.33	**0.00**	0.13	0.78	0.30	0.56	2.48	5.96	3.00
OmniDrive-L	VLA	0.15	0.36	0.70	0.40	0.06	0.27	0.72	0.35	**0.49**	**1.99**	4.86	2.45
VGGDrive	VLA	**0.14**	**0.28**	**0.51**	**0.31**	0.02	**0.10**	0.55	0.22	0.63	2.27	**4.02**	2.31
LECDrive	VLA	**0.14**	0.30	0.53	0.32	0.02	0.12	**0.50**	**0.21**	0.58	2.04	4.17	**2.26**

Table 9: The performance comparison on the OmniDrive dataset [23] focuses on caption-related tasks in autonomous driving scenarios, where base VLMs excel.

Dataset	Metrics	Zero-shot Models		Fine-tuned Models				
		GPT-4o	RoboTron	Ominidrive	Hermes [5]	Baseline	VGGDrive	LECDrive
Omnidrive	BLEU↑	10.91	20.30	38.00	-	37.29	37.58	**37.73**
	CIDEr↑	24.42	34.33	68.60	74.10	86.29	**86.57**	86.34
	ROUGE↑	22.34	23.67	32.60	32.70	34.33	34.40	**34.42**
	Average↑	19.22	26.10	46.40	-	52.64	**52.85**	52.83

DriveLM. This dataset [32] is built upon the real-world driving dataset nuScenes and covers interactive scenarios involving vehicles, pedestrians, and traffic infrastructure on urban roads. The keyframes focus on moments that mark changes in driving intentions, such as acceleration, deceleration, and turning. The data samples not only include "what objects are currently present" (cross-view perception), but also encompass "how objects will move in the future" (action & states prediction), "what the vehicle should do" (planning), "specific behavior classifications" (e.g., fast driving straight, slow right turn), and "trajectory coordinates" (motion). This dataset similarly reflects the model's ability to perform cross-view perception and analyze 3D geometric scene understanding. DriveLM implements four evaluation metrics: accuracy, LLM score, language rule-based evaluation, and match score.

OmniDrive. This dataset [23] is also built upon the nuScenes dataset and primarily focuses on scene description, attention-based question answering, counterfactual reasoning question answering, decision planning question answering, and general dialogue. Due to the limitations in the availability of evaluation data and metrics, we selected scene description and general dialogue from this dataset to train and evaluate the model's ability to understand autonomous driving scene captions. The dataset primarily includes three language evaluation metrics: BLEU, CIDEr, and ROUGE, with the "average" representing the mean of the three metrics.

nuScenes. We conduct open-loop trajectory planning experiments on the challenging public nuScenes dataset [43], which contains 1,000 driving scenes, each lasting approximately 20 seconds. The scene images are captured by six cameras, providing a 360° horizontal field of view (FOV), with keyframes annotated at a frequency of 2Hz. We follow the standard training and testing setup, adhering to the evaluation metrics used in OmniDrive. Among these metrics [23, 43], L2 and Collision Rate are widely used, and we additionally introduce the Intersection Rate, which calculates the rate of collisions or intersections between the predicted trajectories and curbs (road boundaries).

Data Usage Details. For both training and inference, we use multi-view images from the current frame. Specifically, 6-view surrounding images are used for benchmarks built on the nuScenes dataset, including NuInstruct [31], DriveLM [32], OmniDrive [23], and nuScenes-Plan. For NAVSIM, a closed-loop trajectory planning benchmark, we adopt 3-view front-facing images. In trajectory planning tasks on nuScenes-Plan [33] and NAVSIM [34], ego-vehicle states (e.g., velocity and acceleration) and high-level commands (e.g., Go Straight, Turn Left, Turn Right) are incorporated into the prompts.

A.2 Performance Comparison

NAVSIM. As reported in Table 1, the closed-loop trajectory comparison is shown. 1) Compared to AdaThinkDrive, a representative decoupled approach that relies on an additional action decoder for trajectory prediction, LECDrive achieves a **+2.31** improvement in PDMS. 2) Compared with the representative method VGGDrive, which enhances spatial perception by injecting 3D features into VLMs, LECDrive still achieves a **+1.05** gain in PDMS. On the more stringent NAVSIM v2 navtest benchmark (Table 2), LECDrive still achieves a **+0.94** absolute improvement in EPDMS over the representative method VGGDrive. Without exaggeration, LECDrive achieves state-of-the-art performance for VLM-based autoregressive trajectory generation, with a PDMS of 89.81 and an EPDMS of 89.44. These findings highlight that enhancing dense world modeling in VLMs via a progressive latent expert cognitive chain offers a more effective and efficient solution for trajectory prediction than relying solely on auxiliary action decoders.

NuInstruct. Compared to the baseline model (Qwen2.5-VL-7B) fully fine-tuned on the specific dataset (Table 3), the proposed LECDrive achieves substantial improvements on the key MAP and Accuracy metrics, with gains of **+32.10** and **+9.31**, respectively. Language evaluation metrics, such as BLEU, also show consistent improvements, with an average gain of **+12.2** across four metrics. Compared to the representative method VGGDrive, LECDrive further improves MAP and Accuracy metric by **+0.76** and **+0.65**, respectively.

NuScenes-plan. Table 8 presents a quantitative comparison between LECDrive and recent state-of-the-art methods on the nuScenes open-loop trajectory planning benchmark. As shown in the table, LECDrive achieves highly competitive performance compared to the latest approaches. Notably, it demonstrates clear advantages in collision-related metrics and interactions with road boundaries. These improvements can be attributed to our chain-structured dense world modeling design, which enables the model to better perceive both static and dynamic agents.

OmniDrive. Table 9 reports the quantitative results on the OmniDrive language evaluation benchmark. Prior work [23] has shown that transferring VLMs to autonomous driving often introduces a trade-off between language capability and trajectory planning performance. While the base VLM already achieves strong results after fine-tuning on this dataset, our results demonstrate that LECDrive significantly improves driving-related capabilities—including planning, perception, and prediction—without compromising its language performance on captioning tasks.

Visualization Analysis. Figure 4 visualizes the performance of LECDrive across multiple autonomous driving tasks. For trajectory planning, we additionally present the model's internal dense world modeling results, including semantic segmentation and depth estimation generated by dedicated heads, together with the corresponding pseudo-labels provided by expert models. These visualizations show that, with only 8 task-specific latent tokens, LECDrive is able to effectively activate the VLM's capability to understand and model dense world representations. Although a gap remains between the model predictions and expert pseudo-labels, our objective is not to strictly reconstruct the ground-truth dense world, but to enable the VLM to internalize structured dense representations that benefit downstream driving tasks. Moreover, LECDrive also demonstrates satisfactory performance on risk object perception and state prediction tasks.

Effectiveness of the Hybrid Trajectory Formulation. Table 5 also presents an ablation study on the proposed hybrid trajectory formulation, which combines discrete motion primitive retrieval with continuous offset refinement. Compared to the baseline (ID-1), replacing direct coordinate regression with discrete motion primitive retrieval (ID-2) yields a **+1.40** improvement in PDMS, demonstrating the benefit of modeling trajectories in a structured discrete space. However, as shown in Table 1, this formulation alone is not yet competitive with state-of-the-art methods. Adding continuous offset refinement further boosts performance to **88.49** PDMS, making the model competitive with strong expert-designed methods (88.49 *vs.* 88.76 for VGGDrive [25]). These results suggest that discrete primitives provide robustness, while continuous offsets recover fine-grained precision. Overall, the hybrid formulation effectively balances robustness and accuracy, significantly improving trajectory prediction performance.

Effectiveness of the Architectural Variants and Key Hyperparameters. Table 6 presents ablation studies on architectural variants and key hyperparameters of LECDrive. ID-2 removes the task-specific latent tokens used for internalizing multi-expert representations and instead directly relies on the VLM's visual features. Under this setting, significant performance degradation is observed (**-1.08**

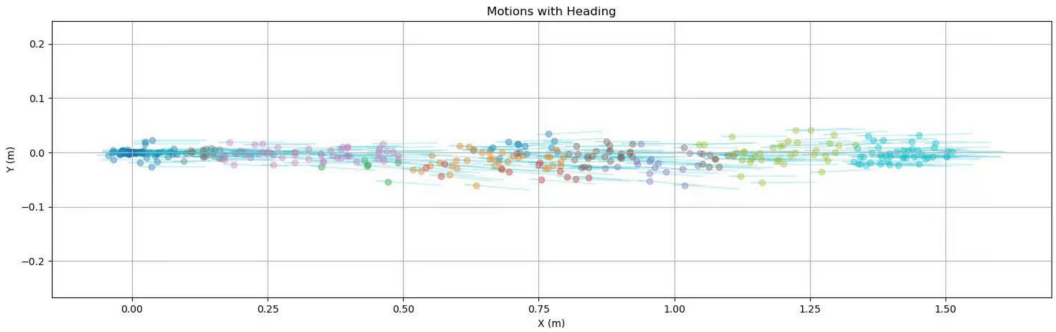


Figure 5: Visualization of clustered motion primitives.

PDMS), indicating that the absence of task-specific latent tokens leads to severe task interference and hinders effective knowledge specialization. ID-3 uses only the final-layer latent tokens without performing multi-layer aggregation of task-specific tokens within the VLM. This configuration results in a performance drop of **-0.55** PDMS, suggesting that explicitly integrating features from earlier layers of the VLM is beneficial for capturing fine-grained representations and improving overall performance. ID-4 and ID-5 further examine the impact of the number of latent tokens and the scale of the base model, showing that both factors affect performance to varying degrees.

A.3 Hybrid Trajectory Formulation and Visualization

For trajectory prediction, we further design a hybrid formulation that combines discrete motion primitive [19] retrieval with continuous offset refinement, improving both the robustness and precision of trajectory planning. Specifically, we first uniformly sample approximately 120K trajectories from the NAVSIM dataset and augment each trajectory with an origin-aligned stationary pose. The pose sequence of each trajectory is then converted into a sequence of vehicle bounding-box polygons according to the vehicle geometry. Based on these polygonal trajectories, we construct a pairwise average Euclidean distance matrix to measure trajectory similarity. A tolerance parameter tol is further introduced to estimate local trajectory density. We subsequently adopt a density-peak-based greedy clustering strategy to iteratively select representative trajectories as cluster centers while removing neighboring samples within the corresponding density region. The process continues until 2048 high-density trajectory centers are obtained or the density threshold reaches its lower bound. Finally, the pose sequences corresponding to these cluster centers are fixed as motion primitives, serving as semantic trajectory atoms for downstream trajectory generation. As illustrated in Figure 5, we visualize the clustered motion primitives used for trajectory generation.

During training, continuous trajectories are reformulated as combinations of discrete motion primitives and continuous offsets for trajectory supervision. Specifically, for each action timestep, a motion primitive is selected from the top-k nearest candidates ranked by trajectory similarity, where $k = 10$ is empirically found to achieve the best performance. Random sampling is further applied within the candidate set to increase multimodal coverage among highly similar local trajectories. After matching the discrete motion primitives to the ground-truth trajectory rollout, the residual errors between the selected primitive and the target trajectory are explicitly computed, including positional and heading deviations. These continuous residuals are then modeled as offset refinement terms to compensate for the information loss introduced by trajectory discretization. As shown in Table 5, the experimental results demonstrate that discrete motion primitives improve trajectory robustness, while continuous offset refinement recovers fine-grained prediction precision. Overall, the proposed hybrid formulation effectively balances robustness and accuracy, leading to substantial improvements in trajectory prediction performance.

Figure 6 visualizes the planning performance of the proposed hybrid trajectory formulation in three representative driving scenarios: going straight, turning left, and turning right. The green trajectory denotes the human-driving ground-truth trajectory, the yellow trajectory represents the prediction decoded solely from discrete motion primitives, and the red trajectory corresponds to the final prediction after continuous offset refinement. As shown in the figure, using only discrete motion primitives often leads to under-steering behavior in turning scenarios and insufficient trajectory smoothness in straight-driving cases, resulting in unstable steering adjustments. In contrast, continuous offset



Figure 6: Visualization of the proposed hybrid trajectory formulation. Green denotes the human-driving ground-truth trajectory, yellow denotes the trajectory decoded solely from discrete motion tokens, and red denotes the final prediction after continuous offset refinement.

refinement effectively improves trajectory quality, leading to smoother, safer, and more accurate driving behaviors.

As illustrated in Figure 7, additional visualizations of closed-loop trajectory planning are further provided. Specifically, three trajectories are visualized in the front-view image to provide an intuitive comparison. In addition, the instruction details used during training on the NAVSIM dataset are presented. The instruction includes the ego-vehicle state information, driving command, and 2-second historical trajectory. The historical trajectory is reconstructed as a combination of discrete motion primitive retrieval and continuous offset refinement ($k = 1$ during reconstruction).



"You are an autonomous driving trajectory prediction system. Predict the ego vehicle's future trajectory for the next 4 seconds (8 steps, 0.5s per step) based on: \n1. Visual perception from front-left, front, and front-right camera views \n2. Historical motion context (last 3 steps): - t-2: <MT1134>+[-0.00, -0.00] - t-1: <MT1134>+[0.01, -0.01] - t-0: <MT742>+[0.00, -0.05] Each <MTxxx> is a motion token representing a discrete motion prototype. The 2D deviation [dx, dy] is a continuous offset applied to the motion token\n3. Steering: 5.91 degrees\n4. Speed: 2.05 m/s\n5. Acceleration: 0.82 m/s²\n6. Active navigation command: Turn Left\nOutput requirements:\n- Predict 8 future motion steps (t+1 to t+8) - Each step must be formatted as: <MTxxx> + [dx, dy] - Use [...] to encapsulate the sequence - Deviation values [dx, dy] should be rounded to two decimal places - Output only the motion sequence, without any extra text"



"You are an autonomous driving trajectory prediction system. Predict the ego vehicle's future trajectory for the next 4 seconds (8 steps, 0.5s per step) based on: \n1. Visual perception from front-left, front, and front-right camera views \n2. Historical motion context (last 3 steps): - t-2: <MT258>+[0.03, -0.00] - t-1: <MT1405>+[-0.02, 0.01] - t-0: <MT1007>+[0.01, -0.04] Each <MTxxx> is a motion token representing a discrete motion prototype. The 2D deviation [dx, dy] is a continuous offset applied to the motion token\n3. Steering: 3.92 degrees\n4. Speed: 6.19 m/s\n5. Acceleration: -0.39 m/s²\n6. Active navigation command: Turn Left\nOutput requirements:\n- Predict 8 future motion steps (t+1 to t+8) - Each step must be formatted as: <MTxxx> + [dx, dy] - Use [...] to encapsulate the sequence - Deviation values [dx, dy] should be rounded to two decimal places - Output only the motion sequence, without any extra text"



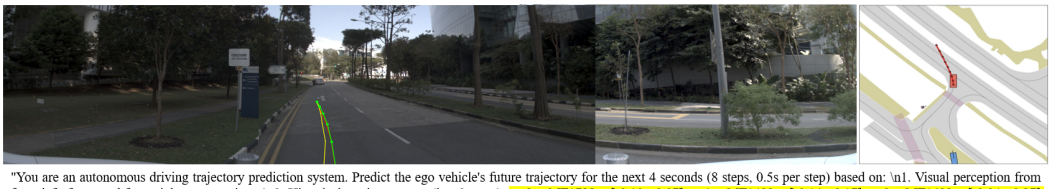
"You are an autonomous driving trajectory prediction system. Predict the ego vehicle's future trajectory for the next 4 seconds (8 steps, 0.5s per step) based on: \n1. Visual perception from front-left, front, and front-right camera views \n2. Historical motion context (last 3 steps): - t-2: <MT1777>+[0.00, 0.03] - t-1: <MT1044>+[0.02, 0.06] - t-0: <MT1551>+[-0.05, 0.07] Each <MTxxx> is a motion token representing a discrete motion prototype. The 2D deviation [dx, dy] is a continuous offset applied to the motion token\n3. Steering: -5.24 degrees\n4. Speed: 5.08 m/s\n5. Acceleration: -0.62 m/s²\n6. Active navigation command: Turn Right\nOutput requirements:\n- Predict 8 future motion steps (t+1 to t+8) - Each step must be formatted as: <MTxxx> + [dx, dy] - Use [...] to encapsulate the sequence - Deviation values [dx, dy] should be rounded to two decimal places - Output only the motion sequence, without any extra text"



"You are an autonomous driving trajectory prediction system. Predict the ego vehicle's future trajectory for the next 4 seconds (8 steps, 0.5s per step) based on: \n1. Visual perception from front-left, front, and front-right camera views \n2. Historical motion context (last 3 steps): - t-2: <MT1056>+[-0.07, 0.04] - t-1: <MT916>+[-0.07, 0.11] - t-0: <MT1863>+[-0.02, 0.13] Each <MTxxx> is a motion token representing a discrete motion prototype. The 2D deviation [dx, dy] is a continuous offset applied to the motion token\n3. Steering: -8.90 degrees\n4. Speed: 4.50 m/s\n5. Acceleration: 0.09 m/s²\n6. Active navigation command: Turn Right\nOutput requirements:\n- Predict 8 future motion steps (t+1 to t+8) - Each step must be formatted as: <MTxxx> + [dx, dy] - Use [...] to encapsulate the sequence - Deviation values [dx, dy] should be rounded to two decimal places - Output only the motion sequence, without any extra text"



"You are an autonomous driving trajectory prediction system. Predict the ego vehicle's future trajectory for the next 4 seconds (8 steps, 0.5s per step) based on: \n1. Visual perception from front-left, front, and front-right camera views \n2. Historical motion context (last 3 steps): - t-2: <MT723>+[0.04, -0.00] - t-1: <MT681>+[-0.01, -0.01] - t-0: <MT342>+[-0.01, -0.03] Each <MTxxx> is a motion token representing a discrete motion prototype. The 2D deviation [dx, dy] is a continuous offset applied to the motion token\n3. Steering: -1.65 degrees\n4. Speed: 5.05 m/s\n5. Acceleration: -1.05 m/s²\n6. Active navigation command: Go Straight\nOutput requirements:\n- Predict 8 future motion steps (t+1 to t+8) - Each step must be formatted as: <MTxxx> + [dx, dy] - Use [...] to encapsulate the sequence - Deviation values [dx, dy] should be rounded to two decimal places - Output only the motion sequence, without any extra text"



"You are an autonomous driving trajectory prediction system. Predict the ego vehicle's future trajectory for the next 4 seconds (8 steps, 0.5s per step) based on: \n1. Visual perception from front-left, front, and front-right camera views \n2. Historical motion context (last 3 steps): - t-2: <MT1728>+[-0.10, -0.07] - t-1: <MT1432>+[-0.14, -0.17] - t-0: <MT1432>+[-0.04, -0.27] Each <MTxxx> is a motion token representing a discrete motion prototype. The 2D deviation [dx, dy] is a continuous offset applied to the motion token\n3. Steering: 17.38 degrees\n4. Speed: 3.03 m/s\n5. Acceleration: 0.51 m/s²\n6. Active navigation command: Turn Left\nOutput requirements:\n- Predict 8 future motion steps (t+1 to t+8) - Each step must be formatted as: <MTxxx> + [dx, dy] - Use [...] to encapsulate the sequence - Deviation values [dx, dy] should be rounded to two decimal places - Output only the motion sequence, without any extra text"

Figure 7: Visualization of the proposed hybrid trajectory formulation in representative autonomous driving scenarios, including left turns, right turns, and straight driving.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly summarize the main contributions of the paper, including the proposed method and its empirical validation. The claims are consistent with the theoretical analysis and experimental results presented in Sections 4 and the Appendix, and the scope and limitations are explicitly discussed.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the limitations of the proposed approach, which are explicitly discussed in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper provides sufficient details to reproduce the main experimental results, including model architecture, training procedures, and evaluation protocols. All information necessary to support the main claims is clearly described in Section 4.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code and data will be made publicly available upon publication, along with detailed instructions for reproducing the main experimental results, including environment setup and execution commands.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the necessary training and evaluation details, including data splits, hyperparameters, optimization settings, and model configurations. These details are described in Section 4, with additional information provided in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper does not report statistical significance measures such as error bars or confidence intervals. Due to the high computational cost of the proposed model, experiments are conducted with fixed random seeds. We demonstrate the effectiveness of the approach through consistent improvements over strong baselines. We acknowledge this limitation and leave a more thorough statistical analysis for future work.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper provides information about the computational resources and training setup for the experiments, as described in Section 4.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in this paper conforms to the NeurIPS Code of Ethics. No ethical concerns have been identified.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: The paper briefly discusses the broader societal impacts in Section 6. The proposed approach enhances VLMs with richer world knowledge through latent expert cognitive chains, which may benefit applications such as VLA-based autonomous driving systems. Potential risks, including misuse or unintended consequences in safety-critical scenarios, are also acknowledged.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not involve the release of high-risk models or datasets, and therefore no specific safeguards are required.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper properly credits the original sources of all datasets, models, and code used. Relevant licenses and terms of use are respected and cited where applicable.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce or release new datasets, models, or other assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects; therefore, IRB approval is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are used only for writing, editing, and formatting purposes, and do not affect the core methodology or scientific contributions; therefore, this question is not applicable.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.