REGAD: RETHINKING REGRESSIVE VERSUS GENERATIVE PLANNING IN END-TO-END AUTONOMOUS DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

End-to-end autonomous driving has recently embraced generative planners to model multimodal future trajectories, yet their iterative sampling process introduces substantial latency, and their empirical gains are often entangled with stronger architectures and anchor priors. This paper revisits whether heavy generative models represent the only solution for autonomous driving and fully exploits the potential of regressive planners. We identify the performance bottlenecks in regressive planners and optimize them accordingly. To this end, we introduce an anchor-refinement framework and a scorer design that incorporates soft scoring labels and a scoring label switching strategy. Finally, we construct three regression variants (RegAD) corresponding to three representative generative planners. Extensive experiments on NAVSIM and nuPlan demonstrate that RegAD matches or surpasses generative planners under the same model complexity while achieving faster training convergence and lower inference latency. Quantitatively, RegAD achieves 90.2 PDMS on NAVSIM with only one-fifth of GoalFlow's latency, and 91.34 non-reactive score on nuPlan with one-eighth of FlowPlanner's latency, both with other modules identical except the planning head. These results suggest that the key design of high-performance multi-modal planning is not necessarily a generative algorithm, but reliable multimodal refinement and candidate scoring. Code will be released upon acceptance.

1 INTRODUCTION

End-to-end (E2E) autonomous driving has emerged as a promising paradigm that unifies traditional sub-tasks (*e.g.*, perception and prediction) to learn driving policies directly from raw sensor inputs Hu et al. (2023); Jiang et al. (2023); Guo et al. (2024); Liao et al. (2025). By bypassing hand-crafted intermediate interfaces, this integration effectively mitigates information loss and compounding errors. Within such unified architectures, the planning module serves as the critical final stage, which translates complex environmental context into safe, efficient, and comfortable trajectories, acting as the decisive bottleneck for the overall driving performance.

Given the central role of the planner, the design of planning policies in E2E autonomous driving has undergone a series of architectural shifts. Early E2E models (*e.g.*, TransFuser Prakash et al. (2021), UniAD Hu et al. (2023), VAD Jiang et al. (2023)) rely on regression-based planning heads that use ego queries to regress a single-mode trajectory in a driving scenario. However, this paradigm struggles to model environmental uncertainty and the multi-modal nature of trajectory planning. Subsequent works attempt to address this by scoring a fixed anchor vocabulary Chen et al. (2024) or processing multiple queries Sun et al. (2024), yet these methods either lack flexibility or suffer from mode collapse. Recently, driven by the success of generative models in robotic policy learning Chi et al. (2025); Black et al. (2024), many E2E models Liao et al. (2025); Xing et al. (2025a); Zheng et al. (2025a); Tan et al. (2025); Wang et al. (2026); Liu et al. (2025) have adopted generative planners to capture the multimodal distribution of driving scenarios, demonstrating SOTA performance and the potential for flexible guidance.

Despite the promising results of generative planners, it is worth revisiting whether complex multi-step generative procedures are always necessary for end-to-end autonomous driving. From a first-

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

principles perspective, generating a driving trajectory that consists of a sequence of simple 2D waypoints is significantly less complex than generating high-resolution images or high-dimensional robotic manipulation policies. Applying complex generative algorithms to a low-dimensional space appears unnecessary from a modeling-difficulty standpoint. Furthermore, since real-world driving datasets typically provide only a single human expert trajectory for a given scenario, purely generative planners often suffer from mode collapse and fail to exhibit the multimodal behaviors as expected Liao et al. (2025). Additionally, the inherent multi-step iterative denoising process incurs high inference latency, conflicting with the real-time demands of autonomous driving systems.

To bridge these fundamental gaps, recent studies have made substantial efforts to adapt generative models to driving tasks. Many methods attempt to accelerate inference and enable multimodal planning by introducing external priors, such as anchor vocabularies Liao et al. (2025) or goal points Xing et al. (2025a), which is equally applicable to simpler regression policies. Meanwhile, some work Wang et al. (2026) employ one-step generation technique Geng et al. (2025) to further compress the inference latency of generative planners. However, these designs introduce an intriguing paradox. As generative models are constrained to minimal denoising steps and become heavily reliant on external priors, their operational mechanism increasingly resembles direct regression.

Therefore, we revisit whether generative models represent the only answer to autonomous driving. We observe that current generative policies Liao et al. (2025); Xing et al. (2025a); Wang et al. (2026) claim superiority over regression by comparing their advanced DiT architectures against the simple MLP heads for TransFuser Liao et al. (2025). This comparison is unfair and does not prove the inherent superiority of diffusion. Additionally, DiffE2E Zhao et al. (2025) achieves SOTA performance by adopting a mixed strategy of diffusion and regression, indicating the underlying superiority of regression methods in certain aspects. We therefore ask: ***Does the empirical success of these generative planners genuinely demonstrate that the generative algorithm itself is more suitable for autonomous driving than regression?***

To answer this question, we investigate the potential of the regression paradigm by exploring the differences between current mainstream generative planners and regression-based planners. To ensure a strictly fair comparison and demonstrate that our improvements stem from fundamental insights rather than extraneous tricks, we build our study upon representative diffusion-based models by directly replacing their generative heads with our proposed regression counterpart.

Specifically, we first introduce a multi-anchor refinement head to enable true multimodal planning within the regression paradigm. While this architectural design significantly enhances both performance and multimodal modeling capabilities, a marginal gap compared to the generative baseline initially remained. Through an in-depth analysis of the candidate trajectories, we uncover a crucial insight that high-quality trajectories produced by the regression paradigm tend to concentrate in fewer modes but actually exhibit a higher performance upper bound. This finding reveals that the primary bottleneck of the regression paradigm lies not in trajectory generation, but in trajectory scoring. To address this, we introduce soft labels and a novel scoring label switching strategy. With these principled designs, our pure regression model surprisingly matches the performance of baseline DiffusionDrive Liao et al. (2025), all while maintaining robust multimodal capabilities. Finally, we demonstrate the broad applicability of our approach by adapting the regression paradigm to other state-of-the-art generative planners like GoalFlow Xing et al. (2025a) and FlowPlanner Tan et al. (2025). As shown in Fig. 1, this adaptation enables the regression paradigm to achieve comparable, and even superior performance with substantially lower latency. Specifically, RegAD achieves a PDMS of 90.2 at only one-fifth of GoalFlow's latency, comparable to the performance of original GoalFlow, and attains a non-reactive score of 91.34 at only one-eighth of FlowPlanner's latency, strongly demonstrating the potential of regressive planners for autonomous driving. In addition, our visualization results suggest that the anchor-based regression paradigm can also produce multimodal plans that cover diverse driving intentions.

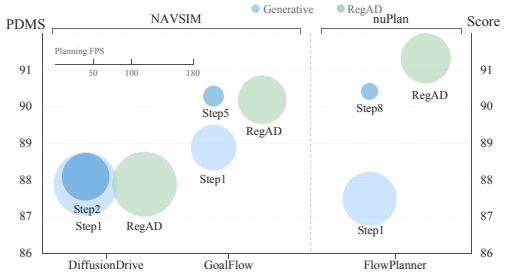

Figure 1: Performance and latency comparison between regressive planners and the generative planners. A larger circle indicates lower latency.

Our contributions are summarized as follows:

- We construct a roadmap from TransFuser to advanced regressive policies, showing that the regression paradigm has modeling potential comparable to generative approaches.
- We identify that the bottleneck of the anchor-refinement framework lies in trajectory selection, and accordingly propose a method based on soft labels and a label-switching strategy, achieving generative-level performance.
- We validate the broad applicability of the regressive paradigm by generalizing it to other generative planners, confirming its robust modeling capacity and efficiency.

2 Related Work

2.1 End-to-End Autonomous Driving

End-to-end autonomous driving aims to integrate perceptionLi et al. (2022); Zhang et al. (2020), prediction Gao et al. (2020), and planning into a unified model that maps sensor observations to driving decisions. Early multi-task models, such as P3 Sadat et al. (2020), TransFuser Prakash et al. (2021), and UniAD Hu et al. (2023), rely on dense BEV representations to support joint scene understanding and planning. Beyond deterministic trajectory regression, another line of work Li et al. (2024) formulates planning as candidate trajectory scoring. Later E2E systems improve either scene representations or model architectures. For representation design, VAD Jiang et al. (2023) introduces vectorized map and agent constraints, GaussianAD Zheng et al. (2024b) represents driving scenes with 3D Gaussians, and SparseAD Zhu et al. (2025) and SparseDrive Sun et al. (2024) use sparse instance queries to model key traffic elements while reducing computation. For architecture and training design, UAD Guo et al. (2024) reduces costly modularization and 3D manual annotation through unsupervised pretext learning, PARA-Drive Weng et al. (2024) designs a parallelized architecture for real-time driving, and FlowAD Guo et al. (2026) models ego-scene interactions with scene flow to enhance downstream driving tasks.

2.2 Generative Models for Planning

Generative models show considerable promise in planning tasks within end-to-end autonomous driving. Early frameworks like GenAD Zheng et al. (2024a) and DiffAD Wang et al. (2025) achieve multi-task joint modeling via VAEs and BEV latent spaces. Subsequent works focus on efficiency and controllability. DiffusionDrive Liao et al. (2025) and MeanFuser Wang et al. (2026) achieve efficient generation via truncated diffusion schedules and a one-step generation algorithm Geng et al. (2025), respectively. GoalFlow Xing et al. (2025a), Mimir Xing et al. (2025b), and GuideFlow Liu et al. (2025) improve physical and intentional compliance through goal conditioning, multi-rate guidance, and joint EBM training, while DiffusionDriveV2 Zou et al. (2025) explores reinforcement learning fine-tuning. Recently, research has evolved toward closed-loop generation and real-world deployment. Diffusion Planner Zheng et al. (2025a) combines the score function with classifier guidance for adaptive planning without rule-based heuristics, and Flow Planner Tan et al. (2025) uses fine-grained tokenization and Flow Matching Lipman et al. (2022) to improve multimodal quality by dynamically reweighting interactions. To transition these models into practice, Hyper Diffusion Planner (HDP) Zheng et al. (2026) systematically validates them on a real-vehicle platform, optimizing the loss space and representation via reinforcement learning post-training to achieve significant performance improvements in complex urban testing.

2.3 Multimodal Trajectory Generation

Autonomous driving requires multimodal trajectory generation because a single scene often admits multiple behaviours. Existing methods can be broadly grouped by how they construct and select candidate trajectories. Static-vocabulary methods predefine a trajectory set and learn to score candidate actions, avoiding iterative generation but being constrained by action-space coverage. Hydra-MDP Li et al. (2024) distills closed-loop metrics over trajectory proposals, WoTE Li et al. (2025) uses a BEV world model for online trajectory evaluation, and SparseDriveV2 Sun et al. (2026) scales static scoring by factorizing trajectories into paths and velocities to obtain a much denser action

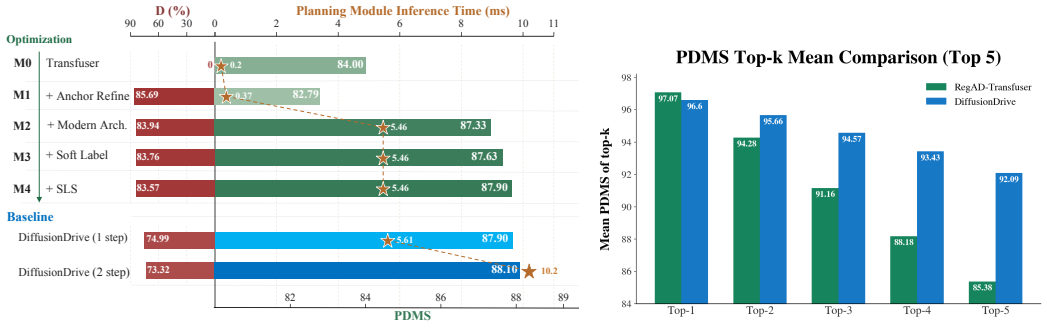

Figure 2: **Analysis of performance gains and remaining bottlenecks in the regressive planner.** Left: We report the evolution of PDMS, the inference latency of the planning module, and the trajectory diversity metric (D) proposed in Liao et al. (2025) as the optimization progresses. Right: Our RegAD-Transfuser achieves a higher top-1 candidate upper bound, but its candidate quality drops faster across lower-ranked modes.

space. NaviHydra Wu et al. (2025) further improves navigation controllability by combining BEV trajectory gathering with a navigation compliance metric, encouraging scored trajectory proposals to follow high-level route commands. In contrast, dynamic generation methods produce scene-conditioned candidates through regression or generative mechanisms. Representative methods such as DiffusionDrive Liao et al. (2025), DiffusionDriveV2 Zou et al. (2025), and GoalFlow Xing et al. (2025a) improve proposal flexibility and diversity, but typically introduce additional generation modules or sampling procedures.

3 METHOD

3.1 OVERVIEW

In this section, we fully explore the planning potential and multi-modal modeling capabilities of regression policies. Our core strategy is to analyze the root causes of the performance gap between regressive and generative policies, and progressively optimize the regression policy based on these insights. We adopt TransFuser Prakash et al. (2021) equipped with a simple MLP planning head as our starting point, utilizing DiffusionDrive Liao et al. (2025) as the comparative baseline. Since DiffusionDrive is constructed by replacing the MLP head of TransFuser with a diffusion-based planning head, we keep the perception module of TransFuser unchanged and optimize only the planning head to ensure a fair comparison. Our progressive roadmap mainly consists of two key phases: Architectural Optimization and Scorer Design. To visualize this process, the left panel of Fig. 2 illustrates the evolution of the PDMS metric, the planning module's inference latency (ms), and Trajectory Diversity Score proposed in Liao et al. (2025) (details in Appendix) on the NAVSIMv1 test set Contributors (2024) across our model variants. The results demonstrate that our method achieves comparable performance to the single-step DiffusionDrive, while exhibiting slightly lower inference latency and superior multi-modal performance.

3.2 ARCHITECTURAL OPTIMIZATION

In this section, for the sake of a fair comparison, we refrain from designing an entirely new architecture, and instead draw as much as possible on existing designs from current generative planners.

Anchor Refine Framework. Our baseline TransFuser ($\mathbf{M_0}$) achieves a PDMS of merely 84.0, where learnable queries interact with environment tokens to generate a single-modal trajectory via an MLP. This contradicts the multimodal nature of trajectories, leading the model to produce "averaged" trajectories that may reside in unsafe regions Chen et al. (2024). To address this, we introduce a trajectory anchor refinement framework like Liao et al. (2025) to enable multimodal trajectory generation in regression models. Specifically, we construct a lightweight anchor vocabulary by clustering trajectories via K-means, same as DiffusionDrive. The model then generates a refined trajectory and a corresponding confidence score for each anchor trajectory:

$$\mathbf{F}^i_{anchor} = \mathrm{CA}(q = \mathrm{PE}(\tau^i_{anchor}), kv = \mathbf{F}_{BEV}) \tag{1}$$

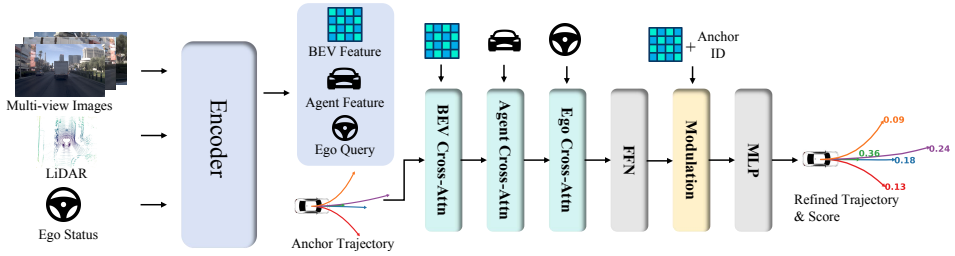

Figure 3: **Architecture of the proposed anchor-refined regression planner.** Multi-view images, LiDAR, and ego status are encoded into scene representations, which interact with anchor trajectories through cross-attention. Anchor-ID-conditioned modulation further distinguishes different trajectory branches, and the planner outputs both refined trajectories and confidence scores for final selection.

$$\tau_i = \text{MLP}(\mathbf{F}^i_{anchor}), score_i = \text{MLP}(\mathbf{F}^i_{anchor}) \tag{2}$$

where $i = 1, 2, ...N_{anchor}$, PE denotes sinusoidal positional encoding and CA refers to the Cross-Attention Module. We employ a Winner-Takes-All (WTA) training strategy, applying trajectory supervision exclusively to the anchor branch closest to the ground truth trajectory, alongside a one-hot score target for this winning branch. This effectively assigns distinct driving intentions to specific anchors, increasing the trajectory diversity score to 85.69, endowing the regression-based method with strong multi-modal generation capability.

Efficient and Context-Rich Interactions. We observe that naively integrating the anchor refine framework counterintuitively degrades the performance score to 82.79 ($\mathbf{M_1}$). This is likely because the auxiliary scoring task demands stronger representational and modeling capacities. Unlike the well-designed network architecture of DiffusionDrive, TransFuser relies on a simple cross-attention module for environmental interaction. In this design, each query interacts indiscriminately with all BEV tokens, which makes it difficult to focus on the most relevant contextual information for accurate trajectory planning and selection. Consequently, we upgrade the architecture of TransFuser by introducing efficient and context-rich environment interaction modules, as illustrated in Fig. 3. Specifically, we replace the Cross-Attention module with a Deformable Attention module Zhu et al. (2020), enabling the model to attend to the environmental context in the vicinity of the ego vehicle's trajectory. Furthermore, we introduce interaction with agent queries and ego status to enrich the planning context following Liao et al. (2025); Zheng et al. (2025b). We also incorporate an anchor-index-conditioned modulation layer to distinguish different anchor branches. Finally, we formulate the prediction as a residual refinement task, significantly alleviating the modeling burden:

$$\tau_i = \text{MLP}(\mathbf{F}^i_{anchor}) + \tau^i_{anchor}, i = 1, 2, ...N_{anchor}. \tag{3}$$

Through these architectural enhancements, the PDMS of our regression model ($\mathbf{M_2}$) increased to 87.33, while maintaining a lower latency and better diversity than DiffusionDrive.

3.3 Scorer Design

Despite the substantial gains brought by the architectural optimizations, our regression model still exhibits a performance gap compared with DiffusionDrive. In principle, the performance of a multimodal planner is jointly determined by candidate quality and selection accuracy. Therefore, we further investigate which factor becomes the main bottleneck for the regression paradigm.

To isolate the trajectory generation capability from the scorer, we analyze the quality distribution of candidate trajectories produced by the regression model and DiffusionDrive. Specifically, for each scene, we evaluate all $N = 20$ candidate trajectories and compute the average PDMS of the oracle Top-k candidates, yielding the Top-k candidate quality distribution shown in the right panel of Fig. 2. Surprisingly, we observe a previously overlooked finding that the upper bound of the regression model is approximately 0.47 PDMS higher than that of DiffusionDrive, indicating that the regression planner already has sufficient capacity to generate high-quality trajectories. However, as k increases, the average PDMS of the regression model drops more sharply, suggesting that its high-performance trajectories are concentrated in fewer modes, whereas the high-quality trajectories of DiffusionDrive are more widely distributed across modes, leading to a slower decline in average PDMS. These results indicate that trajectory scoring, rather than trajectory generation, is the primary bottleneck for the regression paradigm, making a more accurate scorer crucial for fully exploiting its potential.

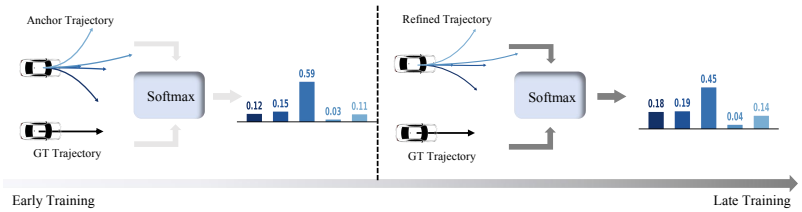

Figure 4: **Illustration of the Scoring Label Switching (SLS) strategy.** During early training, scoring labels are constructed from pre-refinement anchor trajectories to provide stable supervision. During late training, the labels switch to post-refinement trajectories, aligning the scorer with the actual candidates used at inference and enabling more accurate trajectory selection.

Motivated by this observation, we refine the scorer supervision from two aspects. First, we replace rigid one-hot score targets with soft labels to better reflect the continuous quality differences among plausible candidate trajectories. Second, we introduce a scoring label switching strategy to align the scoring labels with the refined trajectories actually used during inference, thereby improving the scorer's ability to identify the best candidate after trajectory refinement.

Soft Scoring Label Supervision. Current methods typically construct a one-hot vector by selecting the anchor closest to the ground truth (GT). Intuitively, this encourages the model to evaluate the suitability of each anchor with the current scene. However, multiple anchors in the same scene may have comparable errors relative to the GT. A rigid one-hot encoding may unfairly penalize equally plausible anchor branches. To address this, we construct soft labels based on the error between different anchors and the GT. We carefully tune a temperature parameter to ensure the confidence distribution remains discriminative while preventing the over-penalization of reasonable anchors. This design improves the model's performance to 87.63 ($\mathbf{M_3}$), without introducing additional inference latency, while its multimodal performance remains superior to that of DiffusionDrive.

Scoring Label Switching Strategy (SLS). We observed that the scorer in DiffusionDrive Liao et al. (2025), the initial scorer in WoTE Li et al. (2025), and the classification labels in our $\mathbf{M_3}$ are determined by calculating the L2 error between the pre-refinement anchors and the GT. Intuitively, this pre-assigns the scene to a specific anchor before refinement. However, the selected anchor may no longer be optimal after the model performs refinement, leading to sub-optimal trajectory selection. A more intuitive approach is to construct selection labels using post-refinement trajectories. Nevertheless, during the early stages of training, post-refinement trajectories generated by a randomly initialized planning head are highly unstable, which can degrade performance. Therefore, we propose a scoring label switching strategy during training (as shown in Fig. 4): for the first half epochs, the model relies on pre-refinement anchors for classification labels to establish basic planning capabilities. For the remaining half epochs, it switches to post-refinement trajectories to foster flexible and accurate classification. This strategy elevates performance to 87.90 ($\mathbf{M_4}$), matching the single-step performance of DiffusionDrive with slightly lower latency and better multimodal performance.

In summary, we find that lightly adapting the advanced framework and model architecture of DiffusionDrive to our regression baseline leads to substantial performance gains. Through further analysis, we identify the trajectory scorer as the primary bottleneck of the regression paradigm, and show that a simple scorer design without introducing any additional parameters can achieve performance comparable to DiffusionDrive. To further ensure a fair comparison, we transfer the same scorer design to DiffusionDrive in the experimental section for comparison.

4 Experiment

To comprehensively validate the performance potential of the regression-based paradigm, we evaluate it across various baselines and datasets. Additionally, we conduct more fine-grained ablation studies on the scorer design and further transfer the same scorer design to DiffusionDrive for a fair performance comparison. Finally, we provide visual analyses of the multimodal planning capability of the regression model, as well as comparisons between the regression planner and the generative planner in a reactive lane-change case.

4.1 BENCHMARKS

NAVSIM. NAVSIM Contributors (2024) is a real-world autonomous driving benchmark built on OpenScene Contributors (2023), a compact redistribution of nuPlan Caesar et al. (2021). It provides surround-view camera, LiDAR, HD map, and object annotation inputs, and focuses on challenging planning scenarios with dynamic driving intentions.

NuPlan. NuPlan Caesar et al. (2021) is a large-scale real-world autonomous driving dataset collected across four cities, covering diverse urban scenarios with complex lane structures and multi-agent interactions. Following Flow Planner Tan et al. (2025), we train the model on the nuPlan 1M training split and evaluate it in closed-loop simulation with an LQR controller. We report results on Val14, Test14-random, and Test14-hard, three nuPlan benchmarks that each cover 14 scenario types. Note that Test14-random involves stochastic scenario selection, which may introduce fluctuations in the evaluation metrics.

Evaluation Metrics For NAVSIM, we use the proposed PDM score (PDMS) Contributors (2024), which combines no at-fault collision (NC), drivable-area compliance (DAC), time-to-collision (TTC), comfort (Comf.), and ego progress (EP). For nuPlan, we mainly focus on closed-loop planning performance under the official benchmark protocols.

4.2 IMPLEMENTATION DETAILS

We compare our regressive planners with representative generative planners and keep the benchmark settings aligned with them. Considering the differences in model architectures and training strategies across existing planners, we design a corresponding regression variant for each baseline. While modifying the planner, we keep the model architecture as consistent as possible for fair comparison. Our selected baselines are as follows:

- **DiffusionDrive** Liao et al. (2025) formulates end-to-end planning as an anchor-based truncated diffusion process, progressively denoising anchor candidates to generate multimodal trajectories. We choose our optimized Transfuser as introduced in Method to compare.
- **GoalFlow** Xing et al. (2025a) introduces a goal-driven flow matching framework, using goal-conditioned generation to model diverse driving intentions and produce multimodal trajectories. To adapt it into a regression variant, we replace the noisy trajectory inputs with learnable queries and substitute the original time embeddings with goal point features. Furthermore, we discard the flow matching formulation and incorporate a modulation layer conditioned on both the goal points and environmental representations. Further details are provided in the Appendix.
- **Flow Planner** Tan et al. (2025) adopts flow matching for trajectory planning with enhanced interactive behavior modeling, improving the planner's ability to handle multi-agent interactions. For the corresponding regression variant, we replace the flow matching algorithm with our multi-anchor refinement framework with a lightweight MLP scoring head. We remove the CFG design and time embedding input to ensure pure regression design. We will introduce more details in the Appendix.

4.3 MAIN RESULTS

We report the main results on NAVSIM in Tab. 1. With slightly lower inference latency, RegAD-Transfuser achieves driving performance comparable to the single-step DiffusionDrive, and yields only a 0.2 PDMS gap relative to the original DiffusionDrive, which requires nearly twice the latency. This indicates the modeling potential of regression-based approaches. Furthermore, compared with GoalFlow, our regression-based method requires only 20% of its inference latency while incurring a negligible 0.1 PDMS performance drop, and outperforms its single-step variant. These results suggest that regression-based methods are also effective within goal-point-guided frameworks.

We further evaluate our method on the nuPlan benchmark, with results summarized in Tab. 2. While requiring only about one-eighth of the inference latency of the generative FlowPlanner, RegAD-FlowPlanner achieves comparable or even superior driving performance. Specifically, it attains the highest scores in Non-reactive (NR) mode across all splits among regression baselines and FlowPlanner, reaching 91.34, 79.01, and 91.10 on Val14, Test14-hard, and Test14, respectively. In Reactive (R) mode, our method remains competitive,

Table 1: Comparison on NAVSIM v1 `navtest` split with closed-loop metrics. *For fair comparison, we use the official scores of versions with one-step generation. **Time** denotes the inference latency of the planning module measured on an NVIDIA GeForce RTX 4090 GPU.

Method	Type	Img. Backbone	NC↑	DAC↑	TTC↑	Comf.↑	EP↑	PDMS↑	Time↓
Transfuser Prakash et al. (2021)	REG	ResNet-34	97.7	92.8	92.8	**100**	79.2	84.0	–
UniAD Hu et al. (2023)	REG	ResNet-34	97.8	91.9	92.9	**100**	78.8	83.4	–
VADv2 Chen et al. (2024)	REG	ResNet-34	97.2	89.1	91.6	**100**	76.0	80.9	–
PARA-Drive Weng et al. (2024)	REG	ResNet-34	97.9	92.4	93.0	99.8	79.3	84.0	–
DRAMA Yuan et al. (2024)	REG	ResNet-34	98.0	93.1	**94.8**	**100**	80.1	85.5	–
ARTEMIS Feng et al. (2025)	REG	ResNet-34	98.3	95.1	94.3	**100**	81.4	87.0	–
DiffusionDrive* Liao et al. (2025)	GEN	ResNet-34	98.3	96.0	94.7	**100**	82.1	**87.9**	5.6 ms
DiffusionDrive Liao et al. (2025)	GEN	ResNet-34	98.2	96.2	94.7	100	82.2	88.1	10.2 ms
RegAD-Transfuser (Ours)	REG	ResNet-34	**98.4**	96.0	94.7	**100**	82.1	**87.9**	**5.5ms**
GoalFlow* Xing et al. (2025a)	GEN	V2-99	**98.4**	97.8	94.1	**100**	84.5	88.9	11.0 ms
GoalFlow Xing et al. (2025a)	GEN	V2-99	98.4	98.3	94.6	100	84.4	90.3	52.9 ms
RegAD-GoalFlow (Ours)	REG	V2-99	98.1	**98.5**	94.5	**100**	**84.7**	**90.2**	**9.7 ms**

Table 2: Performance of regression-based and generative methods on the nuPlan simulation benchmark. Evaluation is conducted in both non-reactive and reactive mode. The highest score for each benchmark is shown in **bold**, while the second-highest score is underlined. In addition, we use * to mark the methods directly leveraging road centerline extracted from the map. **Time** denotes the inference latency of the planning module measured on an NVIDIA H20 GPU.

Type	Planner	Val14		Test14-hard		Test14		Time
		NR	R	NR	R	NR	R	
Expert	Log-replay	93.53	80.32	85.96	68.80	94.03	75.86	–
Learning-based (Generative)	Diffusion Planner Zheng et al. (2025a)	89.87	82.80	75.99	69.22	89.19	**82.93**	–
	Flow Planner Tan et al. (2025)	90.43	**83.31**	76.47	**70.42**	89.88	**82.93**	79.71 ms
	Flow Planner (1midpoint-step) Tan et al. (2025)	87.50	78.44	70.93	67.03	81.12	76.87	16.47 ms
Learning-based (Regression)	PDM-Open*	53.53	54.24	33.51	35.83	52.81	57.23	–
	GameFormer w/o refine.	13.32	8.69	7.08	6.69	11.36	9.31	–
	PlanTF	84.27	76.95	69.70	61.61	85.62	79.58	–
	PLUTO w/o refine.*	88.89	78.11	70.03	59.74	89.90	78.62	–
	RegAD-Flow Planner (Ours)	**91.34**	81.96	**79.01**	69.84	**91.10**	81.51	**9.2 ms**

ranking second on Test14-hard and Test14 with only a marginal gap behind the substantially heavier FlowPlanner. Notably, when compared with a midpoint variant of FlowPlanner at comparable inference latency, RegAD-FlowPlanner outperforms it across all metrics on all splits, indicating a more favorable latency-performance trade-off. In terms of training efficiency, as illustrated in Fig. 5, RegAD-FlowPlanner converges faster, reaching the 90.0 NR score threshold on Val14 in approximately one-third of the epochs required by FlowPlanner. These results indicate that our approach matches the performance of heavier generative models while reducing both training and inference costs.

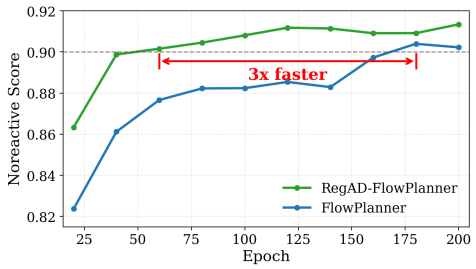

Figure 5: Performance evolution of FlowPlanner and RegAD over training epochs.

Visualization of Multi-modal Planning. To demonstrate RegAD's multimodal planning capability, we visualize predicted trajectories across diverse driving scenarios. Multimodal outputs for FlowPlanner are obtained via inference with five random seeds, while for DiffusionDrive and RegAD, we visualize the five highest-confidence modes. As illustrated in Fig. 6, the trajectories produced by FlowPlanner exhibit limited divergence, with their endpoints remaining relatively concentrated. In contrast, RegAD-FlowPlanner generates trajectories with more clearly distinguishable driving intents, better reflecting the inherent multimodality of driving behavior. For instance, the predicted modes successfully differentiate aggressive from conservative driving styles within the same scene (a), yielding versus proceeding at a crosswalk (b), and going straight versus changing lanes under dense traffic conditions (c)(d). Furthermore, as shown in Fig. 7, RegAD-Transfuser also exhibits intent-level diversity, encompassing both go-straight and lane-change behaviors. The resulting multimodal trajectories are comparable to those of DiffusionDrive and, in certain scenarios, appear even more

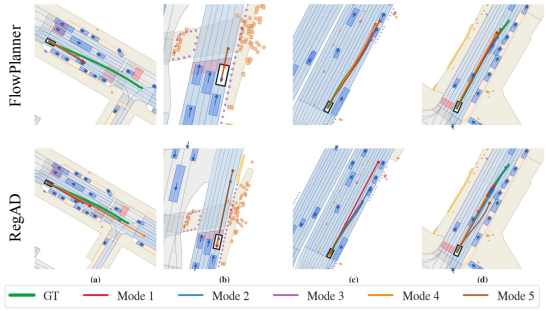

Figure 6: Multimodal trajectory comparison between FlowPlanner and RegAD-FlowPlanner. The thick green curve denotes the GT trajectory, while the remaining thin curves correspond to five predicted modes.

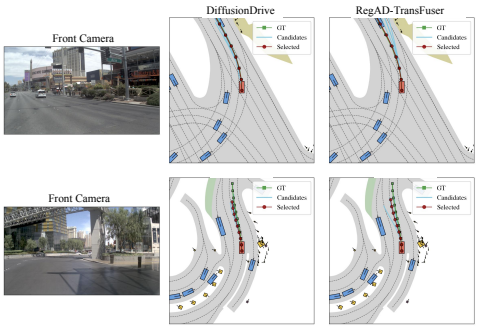

Figure 7: Qualitative visualization of multimodal planning of DiffusionDrive and RegAD-TransFuser.

Table 3: Ablation study on SLS Strategy.

Model	Refined anchor constructing label	Soft label	Switching	NC↑	DAC↑	TTC↑	Com.↑	EP↑	PDMS↑
$\mathcal{M}_2$				98.1	95.8	94.2	**100.0**	81.6	87.3
$\mathcal{M}_2$	✓			98.1	95.6	94.4	**100.0**	81.6	87.3
$\mathcal{M}_2$		✓		98.2	95.7	94.5	**100.0**	81.8	87.6
$\mathcal{M}_2$	✓	✓		98.3	95.7	**94.7**	**100.0**	81.9	87.5
$\mathcal{M}_2$	✓	✓	✓	**98.4**	**96.0**	**94.7**	**100.0**	**82.1**	**87.9**

diverse and plausible. We further provide visualizations of interactive lane-changing closed-loop cases in the Appendix.

4.4 Ablation Studies

Effectiveness of Scorer Design. We ablate our proposed components in Tab. 3. Starting from the baseline $\mathcal{M}_2$, adopting labels constructed by the refined anchor or soft labels individually yields marginal improvements. Even combining both labels only brings the PDMS to 87.5. However, introducing the switching strategy significantly unleashes the model's potential, achieving the highest PDMS of 87.9. This indicates that the switching strategy is essential because it aligns the training process with the actual inference logic.

Adaptation to Generative Models. To ensure a fair comparison, we attempt to adapt our soft label and switching strategy (SLS) to the generative DiffusionDrive. This adaptation leads to a notable performance drop, with PDMS decreasing from 88.1 to 82.4. We attribute this to the noise injection in DiffusionDrive's input trajectories, which disrupts the strict spatial alignment between the input trajectories and the ground-truth score labels. This suggests that our proposed designs are better suited for deterministic regressive planners rather than generative frameworks.

5 Conclusion

This paper revisits whether heavy generative planning is the only effective design choice for end-to-end autonomous driving. We show that a carefully designed regression planner can achieve strong multimodal planning without iterative sampling. Our RegAD combines anchor-based trajectory refinement with soft scoring labels and scoring label switching to improve both candidate generation and selection. Experiments on NAVSIM and nuPlan show that RegAD matches or approaches state-of-the-art generative planners with much higher efficiency. Notably, RegAD-Flow Planner outperforms Flow Planner on nuPlan non-reactive evaluation while using about one-eighth of its inference latency and converging roughly 3× faster. These results highlight that regression-based planning could be a simple and efficient alternative to heavy generative planners. We will discuss the limitations of this work in the Appendix.

AI USE STATEMENT

In this work, we have not used generative AI tools for translation, interpreting results, synthetic data, theory, mathematical claims or proofs, hypotheses, experiment design, method implementation, or dataset cleaning, and qualitative analysis is not applicable to this work. Additionally, we used generative AI tools to aid and polish the writing of the paper. We have reviewed all AI-assisted work. All AI-polished text was carefully checked and revised by the authors to ensure its accuracy and consistency with our intended meaning. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies trajectory-planning methods for autonomous driving using the established NAVSIM and nuPlan research benchmarks. We do not collect new human-subject data or conduct experiments on public roads or real autonomous vehicles. Nevertheless, the safety of autonomous driving systems are critical, and improvements on benchmark metrics should not be interpreted as evidence of deployment readiness. As further discussed in Appendix G, the proposed method may still exhibit failures under long tail traffic events, sensor failures, unusual road configurations, or interactions with vulnerable road users. Any real-world deployment would therefore require substantially broader closed-loop validation, controlled road testing, continuous safety monitoring, and compliance with applicable laws and regulations. We present RegAD solely as a research contribution and do not advocate its direct use in safety-critical systems without such validation.

REPRODUCIBILITY STATEMENT

We provide the information needed to understand and reproduce our experiments throughout the main paper and appendices. Section 4 describes the datasets, evaluation splits, benchmark metrics, comparison protocols, and ablation settings. Appendix A defines the diversity metric and its evaluation protocol. Appendices B and C describe the regression variants of GoalFlow and FlowPlanner, respectively. Appendix D reports the model architectures, training schedules, candidate construction, optimization settings, hardware, and inference configurations. We use the official NAVSIM and nuPlan evaluation protocols and report results across multiple model variants and evaluation splits. The source code and configuration files will be released upon acceptance to facilitate independent reproduction of the reported results.

REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. *arXiv preprint arXiv:2106.11810*, 2021.

Shaoyu Chen, Bo Jiang, Hao Gao, Bencheng Liao, Qing Xu, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. *arXiv preprint arXiv:2402.13243*, 2024.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

NAVSIM Contributors. Navsim: Data-driven non-reactive autonomous vehicle simulation. `https://github.com/autonomousvision/navsim`, 2024.

OpenScene Contributors. Openscene: The largest up-to-date 3d occupancy prediction benchmark in autonomous driving. `https://github.com/OpenDriveLab/OpenScene`, 2023.

Renju Feng, Ning Xi, Duanfeng Chu, Rukang Wang, Zejian Deng, Anzheng Wang, Liping Lu, Jinxiang Wang, and Yanjun Huang. Artemis: Autoregressive end-to-end trajectory planning with mixture of experts for autonomous driving. *IEEE Robotics and Automation Letters*, 11(1):226–233, 2025.

Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11525–11533, 2020.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025.

Mingzhe Guo, Zhipeng Zhang, Yuan He, Ke Wang, and Liping Jing. End-to-end autonomous driving without costly modularization and 3d manual annotation. *arXiv preprint arXiv:2406.17680*, 2024.

Mingzhe Guo, Yixiang Yang, Chuanrong Han, Rufeng Zhang, Shirui Li, Ji Wan, and Zhipeng Zhang. Flowad: Ego-scene interactive modeling for autonomous driving. *arXiv preprint arXiv:2603.13399*, 2026.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *CVPR*, 2023.

Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. *arXiv preprint arXiv:2303.12077*, 2023.

Yingyan Li, Yuqi Wang, Yang Liu, Jiawei He, Lue Fan, and Zhaoxiang Zhang. End-to-end driving with online trajectory evaluation via bev world model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 27137–27146, 2025.

Zhenxin Li, Kailin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Yishen Ji, Zhiqi Li, Ziyue Zhu, Jan Kautz, Zuxuan Wu, et al. Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation. *arXiv preprint arXiv:2406.06978*, 2024.

Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *ECCV*. Springer, 2022.

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

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

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

Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *CVPR*, 2021.

Abbas Sadat, Sergio Casas, Mengye Ren, Xinyu Wu, Pranaab Dhawan, and Raquel Urtasun. Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *ECCV*. Springer, 2020.

Wenchao Sun, Xuewu Lin, Yining Shi, Chuang Zhang, Haoran Wu, and Sifa Zheng. Sparsedrive: End-to-end autonomous driving via sparse scene representation. *arXiv preprint arXiv:2405.19620*, 2024.

Wenchao Sun, Xuewu Lin, Keyu Chen, Zixiang Pei, Xiang Li, Yining Shi, and Sifa Zheng. Sparsedrivev2: Scoring is all you need for end-to-end autonomous driving. *arXiv preprint arXiv:2603.29163*, 2026.

Tianyi Tan, Yinan Zheng, Ruiming Liang, Zexu Wang, Kexin Zheng, Jinliang Zheng, Jianxiong Li, Xianyuan Zhan, and Jingjing Liu. Flow matching-based autonomous driving planning with advanced interactive behavior modeling. *arXiv preprint arXiv:2510.11083*, 2025.

Junli Wang, Yinan Zheng, Xueyi Liu, Zebin Xing, Pengfei Li, Guang Li, Kun Ma, Guang Chen, Hangjun Ye, Zhongpu Xia, et al. Meanfuser: Fast one-step multi-modal trajectory generation and adaptive reconstruction via meanflow for end-to-end autonomous driving. *arXiv preprint arXiv:2602.20060*, 2026.

Tao Wang, Cong Zhang, Xingguang Qu, Kun Li, Weiwei Liu, and Chang Huang. Diffad: A unified diffusion modeling approach for autonomous driving. *arXiv preprint arXiv:2503.12170*, 2025.

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

Hanfeng Wu, Marlon Steiner, Michael Schmidt, Alvaro Marcos-Ramiro, and Christoph Stiller. Navihydra: Controllable navigation-guided end-to-end autonomous driving with hydra-distillation. *arXiv preprint arXiv:2512.10660*, 2025.

Zebin Xing, Xingyu Zhang, Yang Hu, Bo Jiang, Tong He, Qian Zhang, Xiaoxiao Long, and Wei Yin. Goalflow: Goal-driven flow matching for multimodal trajectories generation in end-to-end autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 1602–1611, 2025a.

Zebin Xing, Yupeng Zheng, Qichao Zhang, Zhixing Ding, Pengxuan Yang, Songen Gu, Zhongpu Xia, and Dongbin Zhao. Mimir: Hierarchical goal-driven diffusion with uncertainty propagation for end-to-end autonomous driving. *IEEE Robotics and Automation Letters*, 11(2):2178–2185, 2025b.

Chengran Yuan, Zhanqi Zhang, Jiawei Sun, Shuo Sun, Zefan Huang, Christina Dao Wen Lee, Dongen Li, Yuhang Han, Anthony Wong, Keng Peng Tee, et al. Drama: An efficient end-to-end motion planner for autonomous driving with mamba. *arXiv preprint arXiv:2408.03601*, 2024.

Zhipeng Zhang, Houwen Peng, Jianlong Fu, Bing Li, and Weiming Hu. Ocean: Object-aware anchor-free tracking. In *European conference on computer vision*, pp. 771–787. Springer, 2020.

Rui Zhao, Yuze Fan, Ziguo Chen, Fei Gao, and Zhenhai Gao. Diffe2e: Rethinking end-to-end driving with a hybrid action diffusion and supervised policy. *arXiv preprint arXiv:2505.19516*, 2025.

Wenzhao Zheng, Ruiqi Song, Xianda Guo, Chenming Zhang, and Long Chen. Genad: Generative end-to-end autonomous driving. In *European Conference on Computer Vision*, pp. 87–104. Springer, 2024a.

Wenzhao Zheng, Junjie Wu, Yao Zheng, Sicheng Zuo, Zixun Xie, Longchao Yang, Yong Pan, Zhihui Hao, Peng Jia, Xianpeng Lang, et al. Gaussianad: Gaussian-centric end-to-end autonomous driving. *arXiv preprint arXiv:2412.10371*, 2024b.

Yinan Zheng, Ruiming Liang, Kexin Zheng, Jinliang Zheng, Liyuan Mao, Jianxiong Li, Weihao Gu, Rui Ai, Shengbo Eben Li, Xianyuan Zhan, et al. Diffusion-based planning for autonomous driving with flexible guidance. *arXiv preprint arXiv:2501.15564*, 2025a.

Yinan Zheng, Tianyi Tan, Bin Huang, Enguang Liu, Ruiming Liang, Jianlin Zhang, Jianwei Cui, Guang Chen, Kun Ma, Hangjun Ye, et al. Unleashing the potential of diffusion models for end-to-end autonomous driving. *arXiv preprint arXiv:2602.22801*, 2026.

Zhiyu Zheng, Shaoyu Chen, Haoran Yin, Xinbang Zhang, Jialv Zou, Xinggang Wang, Qian Zhang, and Lefei Zhang. Resad: Normalized residual trajectory modeling for end-to-end autonomous driving. *arXiv preprint arXiv:2510.08562*, 2025b.

Runwen Zhu, Jianbo Zhao, Diankun Zhang, Guoan Wang, Xiwu Chen, Siyu Zhang, Jiahao Gong, Qibin Zhou, Wenyuan Zhang, Ningzi Wang, et al. Sparsead: Sparse query-centric paradigm for efficient end-to-end autonomous driving. *IEEE Transactions on Artificial Intelligence*, 2025.

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

Jialv Zou, Shaoyu Chen, Bencheng Liao, Zhiyu Zheng, Yuehao Song, Lefei Zhang, Qian Zhang, Wenyu Liu, and Xinggang Wang. Diffusiondrivev2: Reinforcement learning-constrained truncated diffusion modeling in end-to-end autonomous driving. *arXiv preprint arXiv:2512.07745*, 2025.

APPENDIX

Technical appendices provide additional definitions and implementation details for the methods and experiments described in the main paper.

Appendix Contents.

- Appendix A: Diversity Score
- Appendix B: Regression Variant of GoalFlow
- Appendix C: Regression Variant of Flow Planner
- Appendix D: Additional Implementation Details
- Appendix E: Case Study
- Appendix F: Limitations
- Appendix G: Potential Societal Impact

A DIVERSITY SCORE

This section details the diversity score used in Fig. 2 in the main paper. We adopt the area-overlap based metric introduced by DiffusionDrive Liao et al. (2025), but use it here as a diagnostic tool for comparing the multi-modal coverage of generative and regressive planners under the same candidate budget.

A.1 METRIC DEFINITION

For a given scene, suppose a planner outputs N trajectory candidates $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_N\}$. We rasterize each candidate into its swept spatial region and compare every candidate with the region covered by the whole candidate set. Let $\bigcup_{j=1}^{N} \tau_j$ denote this overall covered region. The diversity score is computed as

$$\mathcal{D} = 1 - \frac{1}{N} \sum_{i=1}^{N} \frac{\mathrm{Area}\left(\tau_i \cap \bigcup_{j=1}^{N} \tau_j\right)}{\mathrm{Area}\left(\tau_i \cup \bigcup_{j=1}^{N} \tau_j\right)}. \tag{4}$$

Here, $\mathrm{Area}(\cdot)$ denotes the area of a rasterized trajectory region. The IoU term measures how much a single candidate overlaps with the support spanned by all candidates. If all candidates follow nearly the same path, each individual region almost coincides with the union region, producing a large mean IoU and a small $\mathcal{D}$. In contrast, candidates that spread across different maneuvers enlarge the union region while keeping each individual overlap relatively small, leading to a larger diversity score.

A.2 EVALUATION PROTOCOL

For diffusion-based planners, the candidate set is obtained by drawing 20 initial noise samples and running the corresponding denoising procedure. For our regression-based planners, the same number of candidates is obtained from the deterministic anchor-refinement head. We evaluate the score scene by scene on the NAVSIM v1 test split and average it over the target split.

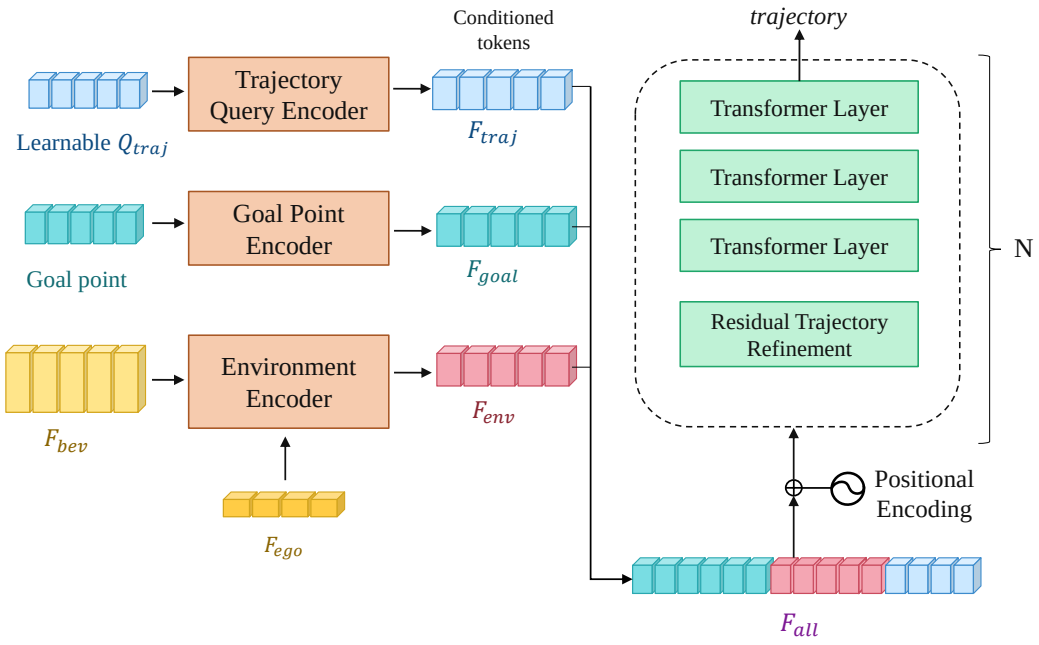

Figure 8: Architecture of RegAD-GoalFlow.

B REGRESSION VARIANT OF GOALFLOW

This section describes how we adapt GoalFlow Xing et al. (2025a) into a regression-based planner while keeping its goal-guided planning structure as consistent as possible. GoalFlow follows a goal-guided generative planning architecture. It first encodes the camera-LiDAR observations and ego status into scene-level planning features, and then uses a goal-point branch to represent the intended driving destination. In its trajectory planner, noisy trajectory samples are encoded together with the scene and goal conditions, while a denoising-time embedding specifies the current flow-matching step. The planner predicts flow updates for these trajectory samples and obtains future trajectories through iterative sampling at inference time.

As shown in Fig.8,to build the regression variant, we keep the goal-guided structure of GoalFlow and only adapt the trajectory planning branch. From the model-architecture perspective, we replace the noisy trajectory inputs with learnable queries, where each query corresponds to one future pose in the 8-step planning horizon. These queries serve as deterministic planning tokens, so the model can maintain a structured representation of future motion in one forward pass. We also replace the denoising-time embedding with goal-point features. In the oracle-navigation setting, the goal feature is derived from the terminal point of the supervised planning horizon. In the student-navigation setting, we reuse the vocabulary of candidate goal points selected by the saved imitation and DAC scores and encode it with the same 2D positional embedding as in GoalFlow. This design makes the regression branch condition directly on the intended destination represented by the goal point.

After obtaining the scene token, the optional goal-point token, and the learnable trajectory queries, we process them as a unified token sequence and refine the trajectory tokens with residual updates in the trajectory latent space. A lightweight modulation layer is used inside the residual refinement step, conditioned on the pooled environment representation and the goal-point feature. The modulation applies a gated feature-wise adjustment to the trajectory tokens, which provides a simple mechanism for injecting scene and goal information into the regression update.

During the training phase, we discard the flow-matching target between noisy samples and the ground-truth trajectory, and instead directly supervise the predicted future trajectory with an L1 regression loss. The auxiliary BEV semantic loss and optional agent detection losses are kept consistent with GoalFlow. At inference time, the regression variant outputs the planned trajectory after a single forward pass, without denoising timesteps, trajectory-noise expansion, or iterative sampling. This

conversion preserves the goal-guided planning interface while matching the training objective and inference procedure to a deterministic regression planner.

C Regression Variant of Flow Planner

This section describes the regression counterpart of Flow Planner Tan et al. (2025). The goal is to isolate the effect of the generative flow matching algorithm while preserving the surrounding architecture and interaction modeling as much as possible.

We start from the original Flow-Planner implementation, which follows an encoder-decoder planning architecture. The encoder first encodes heterogeneous scene elements, including surrounding agents, lanes, static obstacles, and the navigation route, into scene token representations. Then the ego-vehicle's future trajectory is decomposed into overlapping segments and later encoded as a sequence of action tokens. Next, the decoder uses attention-based fusion to condition these trajectory tokens on the encoded scene context. In the original FlowPlanner, the decoder is trained with a flow-matching objective, where the model predicts the corresponding target velocity field given an intermediate trajectory state along a linear interpolation path. During inference, a four-step mid-point ODE solver is used to integrate the learned dynamics from an initial noise sample to the final trajectory.

As shown in Fig.9, our modifications are designed to preserve the original network architecture as much as possible. Specifically, the original scene encoder, trajectory token decoder, attention modules, route conditioning, action-token representation, and trajectory stitching procedure are all retained. At the architectural level, we only introduce a fixed trajectory anchor library and add a lightweight scoring head on top of the decoder trajectory-token representations. The anchor library provides a set of representative future motion patterns, while the scoring head predicts the confidence score for each anchor-conditioned trajectory candidate. Apart from these additions, we do not introduce any extra scene encoder, interaction module, or increase the depth or width of the backbone.

The main changes lie in the training and inference algorithms. We no longer train the model to denoise noisy trajectories using flow matching, nor do we rely on an ODE solver for iterative trajectory generation. Instead, we reformulate generation as one-step anchor-based residual regression. Given the scene context, each clean anchor trajectory is directly fed into the original decoder, which predicts a residual correction relative to that anchor. The final candidate trajectory is obtained by adding the predicted residual to the anchor. During training, the ground-truth trajectory is assigned to the nearest anchor, and a winner-takes-all regression loss is applied only to the matched candidate. Meanwhile, the scoring head is trained with a focal loss to identify the correct motion mode among all anchor candidates. During inference, all anchors generate candidate trajectories in parallel within a single forward pass, and the candidate with the highest score predicted by the scoring head is selected as the final planning output.

Overall, this version should be understood primarily as an algorithmic modification to the trajectory generation objective and sampling procedure, rather than a redesign of the core Flow-Planner network architecture. It replaces the original stochastic flow-matching and ODE-based sampling process with deterministic anchor-conditioned residual regression and score-based candidate selection, while preserving the core encoder-decoder backbone of Flow-Planner.

D Additional Implementation Details

RegAD-Transfuser. Our model backbone adopts a ResNet-34 encoder, where image and LiDAR features are processed separately and then fused through a multi-scale Transformer to obtain the BEV representation. For trajectory prediction, we use 20 trajectory anchors obtained by offline K-means clustering. The prediction horizon is 4 seconds with a sampling interval of 0.5 seconds, resulting in 8 future poses for each trajectory. During training, soft classification labels are constructed from the distance between the anchors and the ground-truth trajectory for the first 50 epochs. After 50 epochs, they are instead constructed from the distance between the refined predicted trajectories and the ground truth. The classification soft-label temperature is changed from 0.25 to 0.05 accordingly. Our model is trained from scratch on the navtrain split for 100 epochs using the AdamW optimizer on 8 NVIDIA H20 GPUs, with a total batch size of 64 and a learning rate of 6e-4.

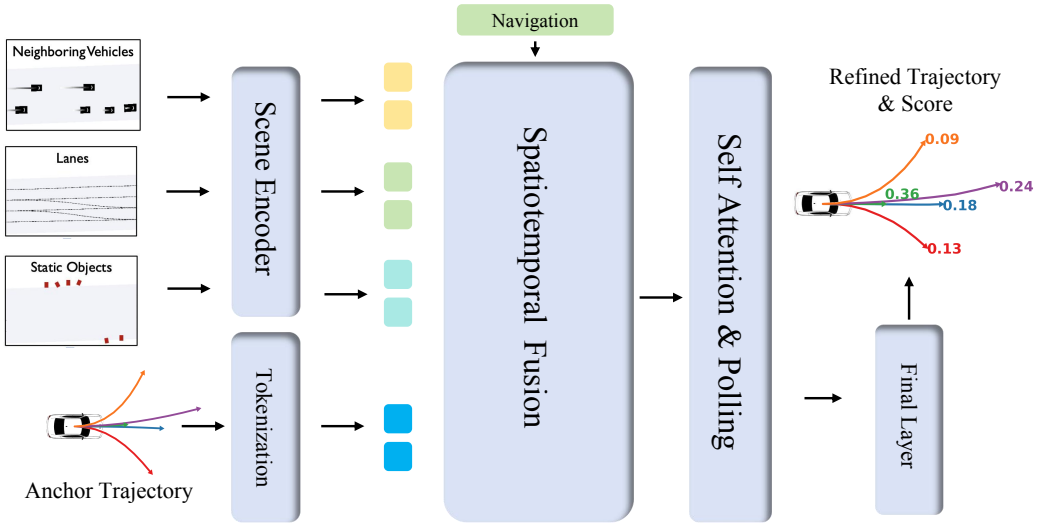

Figure 9: Architecture of RegAD-FlowPlanner.

RegAD-GoalFlow. The training is conducted in three stages on the NAVSIM trainval split. We first train the perception module from V99 pretrained weights, fusing stitched 2048×512 front-view images and LiDAR with a V-99-eSE image backbone, a ResNet-34 LiDAR encoder, and a multi-scale Transformer to produce an 8×8 BEV representation. We then train the trajectory planner for 100 epochs with Adam using a learning rate of 1e-4, a StepLR schedule with step size 20 and gamma 0.8, and batch size 64 on 8 NVIDIA RTX 4090 GPUs. The trajectory, agent classification, agent box regression, and BEV semantic losses are weighted by 50.0, 0.2, 0.05, and 0.2, respectively. Finally, the goal-point module is trained for 100 epochs with a batch size of 16 on an 8192-point vocabulary, with BEV semantic, DAC-score, and imitation-score losses weighted by 10.0, 0.005, and 1.0.

RegAD-FlowPlanner. The training is conducted on 8 NVIDIA H20 GPUs using the 1M training split of nuPlan. The model is trained for over 200 epochs with a batch size of 2048. We use the AdamW optimizer with a learning rate of 5e-4. In terms of model architecture, the scorer head is implemented as a simple single-layer MLP, and the number of trajectory anchors is set to 20. The overall training objective is a weighted sum of the planning regression loss, the mode classification loss, and the overlapping-segment consistency loss, with weights of 1.0, 0.5, and 0.5, respectively. All other implementation details follow FlowPlanner.

E Case Study

We present a closed-loop lane-change case comparing FlowPlanner and RegAD, as shown in Fig. 10. At frame 47, both methods initiate a left lane-change maneuver. By frame 94, FlowPlanner continues to encroach into the adjacent left lane, whereas RegAD, recognizing that the maneuver can no longer be safely completed, promptly adjusts its trajectory and aborts the lane change, reverting to the original lane. Consequently, FlowPlanner collides with the vehicle in the adjacent lane, while RegAD safely returns to its original lane and continues driving. This case suggests that RegAD also exhibits a certain degree of interaction-aware responsiveness under closed-loop settings.

F Limitations

Although our experiments support the effectiveness of regression-based planning, the current validation remains limited in scope. Evaluating more datasets and baseline planners would further test the generality of our conclusions across different data distributions and model designs. In addition, due to constraints in computational resources and experimental cost, we have not yet evaluated the proposed method on larger-scale datasets or real autonomous driving platforms, which will be an important direction for future work.

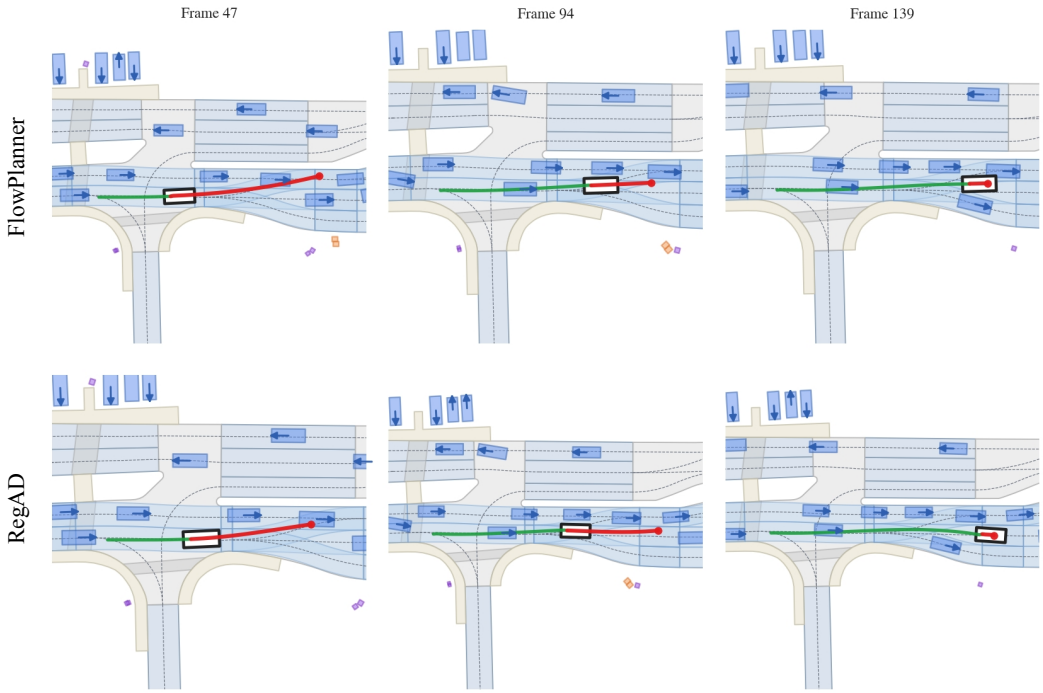

Figure 10: Comparison between FlowPlanner and RegAD-FlowPlanner in a lane-changing case.

G Potential Societal Impact

This work studies regression-based trajectory planning for autonomous driving, with the goal of improving planning efficiency while maintaining competitive driving performance. If deployed responsibly, more efficient planning models may reduce the computational cost of autonomous driving systems and make real-time decision-making more accessible on resource-constrained platforms. This could contribute to safer and more reliable driver-assistance or autonomous mobility systems, especially in settings where fast response is important.

At the same time, improvements in planning models may also introduce societal risks if they are over-interpreted or deployed without sufficient validation. Benchmark gains do not guarantee safety in open-world traffic, where long-tail events, sensor failures, unusual road layouts, and interactions with vulnerable road users can expose failure modes that are not fully captured by standard datasets. Therefore, the proposed method should be regarded as a research contribution rather than a complete deployment-ready driving system. Any real-world use would require extensive closed-loop simulation, controlled road testing, safety monitoring, and compliance with local regulations.