Unveiling the Surprising Efficacy of Navigation Understanding in End-to-End Autonomous Driving

Zhihua Hua[1,3], Junli Wang[4,3], Pengfei Li[3], Qihao Jin[1,3], Bo Zhang[2], Kehua Sheng[2], Yilun Chen[3], Zhongxue Gan[1], and Wenchao Ding[1]

***Abstract*— Navigation information serves as a critical component in end-to-end autonomous driving systems, providing essential decision-making references for planner. However, our experimental results reveal that many existing end-to-end autonomous driving systems may not adequately comprehend navigation information, consequently failing to execute appropriate planning based on navigation information. To overcome this limitation, we propose a Sequential Navigation Guidance (SNG) framework, which is designed based on real-world navigation patterns. The SNG incorporates both a navigation path to constrain long-term trajectories and Turn-by-Turn (TBT) information for real-time decision logic. We also introduce an efficient and streamlined model that achieves state-of-the-art (SOTA) performance solely through the accurate modeling of navigation information, without requiring auxiliary loss functions from perception tasks. Codes will be publicly available at `https://github.com/Zhihua-Hua/NavigationDrive`.**

I. Introduction

In recent years, end-to-end autonomous driving system has garnered significant attention from researchers [1], [2]. The end-to-end paradigm simplifies traditional modular systems, is better suited to data-driven training approaches, and demonstrates enhanced generalization performance [3], [4].

Navigation information plays a pivotal role in end-to-end autonomous driving systems [5], providing essential directional references for trajectory planning. Unlike prediction [6], which generates multimodal trajectory forecasts, planning requires explicit navigation inputs to produce deterministic driving trajectories [7], [8]. Despite the critical role of navigation information in end-to-end autonomous driving systems, we have made a surprising observation: *removing or corrupting navigation information in existing end-to-end driving methods minimally affects planning performance and, in some cases, even improves specific performance metrics*. For instance, as illustrated in Fig. 1, our experiments with the Transfuser [9] on the NAVSIM [10] benchmark demonstrate that complete removal of navigation information paradoxically yields superior results. This phenomenon contradicts basic driving logic, as one would anticipate a substantial decline in planner performance when explicit navigation information is absent. This unexpected outcome raises a critical question: *Do current end-to-end autonomous driving systems truly understand and utilize navigation information?*

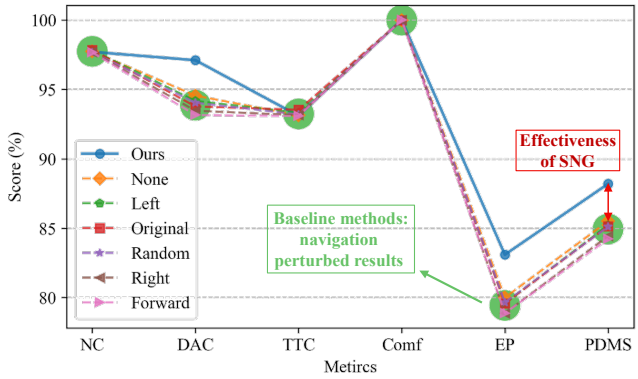

Fig. 1: We demonstrate the effects of introducing perturbations to the driving command and the effectiveness of the SNG. In driving command based method (represented by the dotted line), the complete removal or the introduction of a certain level of noise has minimal impact on the planning outcomes, as all experimental metrics converge within the green circle. In contrast, the model based on SNG (represented by the solid line) demonstrates significant improvements in DAC, EP and PDM scores [10], which are closely related to the quality of navigation information.

Our answer is unequivocally negative. Current research [11]–[13] predominantly employs driving commands (such as Turn Left, Go Forward, Turn Right, None) to represent navigation information, utilizing one-hot encoding to discretize driving behaviors into finite categories. However, this approach exhibits the following limitations: (1) the annotation process relies on a fixed temporal horizon or spatial intervals [7], [10] , which can lead to ambiguous interpretations in certain scenarios. As illustrated in Fig. 2 (a), a vehicle's stationary trajectory caused by a front vehicle is erroneously labeled as "Go Forward", while the actual navigation information is "Turn Left"; (2) this representation suffers from oversimplification. In real-world scenarios, navigation information is influenced by a complex interplay of road conditions, traffic regulations, and surrounding vehicle dynamics [14], rendering the limited discrete categories insufficient for comprehensive representation. As shown in Fig. 2 (b), where a rule-based "Turn Right" command significantly deviates from the vehicle's actual

[1] Academy for Engineering and Technology, Fudan University, China {zhhua24, qhjin24}@m.fudan.edu.cn, {ganzhongxue, dingwenchao}@fudan.edu.cn
[2] Didi Chuxing {zhangbo, shengkehua}@didiglobal.com
[3] Institute for AI Industry Research (AIR), Tsinghua University, China {li-pf22}@mails.tsinghua.edu.cn, {chenyilun}@air.tsinghua.edu.cn
[4] Institute of Automation, Chinese Academy of Sciences {wangjunli2022}@ia.ac.cn

continuous lane-changing behavior. Consequently, end-to-end autonomous driving systems based on driving commands fail to effectively utilize navigation information, and their performance likely stems from overfitting to specific input channels [15], [16].

To address these limitations and enhance the navigation semantic comprehension capabilities of end-to-end autonomous driving systems, we propose a novel paradigm of Sequential Navigation Guidance (SNG), inspired by real-world navigation patterns [17]. The SNG effectively represents navigation information by integrating static global path planning with dynamic high-level guidance: (1) Navigation Path: A predefined trajectory segment extracted from the global path, serving as a reference line for planning, which can be easily sampled from coarse-grained navigation points obtained through the SD map; (2) Real-time Turn-by-Turn (TBT) Information: A comprehensive set of high-level guidance cues, including current route instructions, distance and time estimations, traffic conditions, and speed limits, which collectively inform the planning process. Both types of information can be conveniently acquired through navigation APIs and are readily available off-the-shelf in practical deployment.

The proposed SNG demonstrates remarkable efficacy in modeling navigation information, offering a plug-and-play solution that significantly enhances the planning capabilities of end-to-end autonomous driving systems. We have developed a streamlined and efficient pipeline that, without the need for auxiliary loss from perception tasks, achieves state-of-the-art performance on both the Bench2Drive [18] a closed-loop benchmark based on Carla [19] and the NAVSIM [10] a real-world evaluation benchmark. Our contributions can be summarized as follows:

1. To address the limitations of current navigation representation, we propose a novel Sequential Navigation Guidance approach to structure navigation information, offering both long-term trajectory constraints and real-time decision-making logic.
2. We investigate the optimal combination of navigation paths and TBT information, demonstrating that efficient modeling of navigation information can be achieved without relying on high-precision navigation paths.
3. We develop an effective pipeline that, without incorporating auxiliary tasks and utilizing precise navigation information, achieves state-of-the-art (SOTA) performance in both Bench2Drive and NAVSIM benchmarks.

II. Related Work

A. End to end autonomous driving

Traditional autonomous driving systems are often composed of multiple modular components [20]–[22], whereas end-to-end autonomous driving enables a direct mapping from raw sensor data to planning trajectories [3], [23]. A subset of research [24]–[26] focuses on simulator-based [19] closed-loop end-to-end autonomous driving. While predefined routes can be obtained within simulation environments,

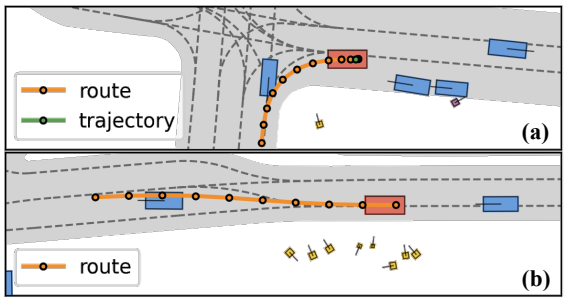

Fig. 2: We demonstrate the limitations of driving commands in representing navigation information. (a) The vehicle generates a stationary trajectory due to yielding to a preceding vehicle. The driving command annotated based on the trajectory is “Go Forward”, while the actual driving path of the vehicle is “Turn Left”. (b) The vehicle’s driving path involves a lane change to the right, but the annotated driving command is “Turn Right”.

some studies [24], [25] continue to employ driving commands as a representation of navigation information. Due to the significant gap between simulation and the real world, open-loop end-to-end autonomous driving based on real-world scenarios [27] remains the primary focus of most research efforts, particularly those utilizing driving commands. UniAD [3] has significantly enhanced the performance of autonomous driving systems by integrating multiple modules into an end-to-end framework. VAD [7] employs a fully vectorized approach to model driving scenarios, ensuring planning safety while improving operational efficiency. BEVPlanner [16] transforms sensor inputs into BEV (Bird’s Eye View) features, serving as an intermediate representation within the end-to-end architecture. GenAD [13] introduces a novel generative framework that aids planning tasks by predicting the dynamic interactions between the ego vehicle and the environment.

B. Navigation information for planning

Navigation information plays a critical role in autonomous driving planning. Current end-to-end benchmarks primarily rely on driving commands as navigation inputs. In real-world datasets [27]–[29], driving commands (e.g., “Turn Left”) are implicitly inferred from expert trajectories to model navigation information. Although simulators [18], [19], [30] provide waypoints between the current position and the target location, they still use discrete driving commands as the primary input. Several methods, such as UniAD [3] and BEVPlanner [16], embed navigation commands into latent spaces as additional model inputs. ST-P3 [8] samples multiple trajectories and filters them based on geometric features aligned with driving commands, while VAD [7] generates results for all command categories and selects the corresponding trajectory as the final output. TCP [24] and TransFuser [10], [9] concatenate driving commands with ego states as conditional inputs. However, relying solely on

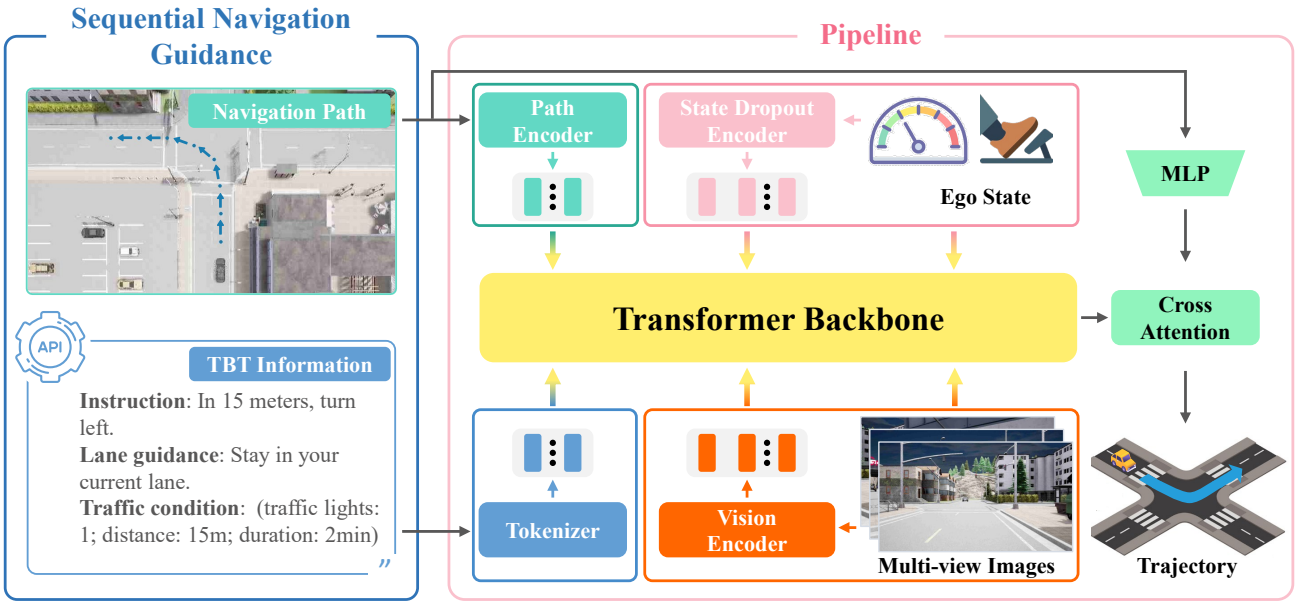

Fig. 3: Overview of our method. Sequential navigation guidance is consists of navigation path and TBT information. The full pipeline of our model is divided into two phases. The multimodal feature fusion encoder and the transformer backbone of LLM (Large Language Model).

driving commands to model navigation information leads to issues such as intent ambiguity (e.g., a left lane change may be misclassified as both "Go Straight" and "Turn Left") and significant deviations from real-world scenarios. To address these limitations, we propose integrating TBT (Turn-by-Turn) instructions and navigation paths to model more accurate navigation information, thereby improving planning rationality, safety, and human-like interaction.

C. Multimodal large models for planning

Multimodal large models facilitate seamless interaction and understanding across diverse data types, driving transformative innovations in fields such as natural language processing and beyond [31]–[33]. Given the necessity of processing sensor data from multiple modalities and performing joint planning in autonomous driving systems, multimodal large models naturally serve as an effective backbone for such systems. DriveGPT4 [34] leverages multimodal large models to process multi-frame video and text inputs, simultaneously outputting reasoning processes during planning, thereby significantly enhancing the interpretability and interactivity of autonomous driving systems. LMDrive [35] unifies multimodal sensor data into a textual feature space, greatly improving the interactivity of autonomous driving systems and demonstrating exceptional performance in CARLA [19] closed-loop evaluation. DriveLM [12] mimics human reasoning processes by structuring driving as a graph-structured reasoning task, achieving notable improvements in planning effectiveness and zero-shot capabilities. Therefore, we propose an efficient pipeline based on a multimodal large model, capable of processing data from various sensor modalities and performing end-to-end planning tasks.

III. System Overview

As illustrated in Fig. 3, our approach comprises two key components: Sequential Navigation Guidance (SNG) and the network pipeline. The SNG integrates a navigation path that imposes long-term trajectory constraints, along with Turn-by-Turn (TBT) information that facilitates real-time decision-making logic. Both types of information are easily accessible through navigation APIs in real-world deployments, significantly reducing the sim-to-real gap. The network pipeline consists of a multimodal feature fusion encoder and a transformer decoder. The encoder processes and integrates multimodal data, including text, navigation points, ego-vehicle states, and multi-view images, and projects these features into the latent space of the transformer decoder. The transformer decoder, which includes a transformer backbone and a cross-attention decoder, takes the fused features as input and generates hidden states. These states undergo cross-attention with navigation path features to regress the predicted trajectory.

IV. Methods

We first introduce the modeling approaches for navigation information, including the rule-based approach in Section IV-A.1 and sequential navigation guidance in Section IV-A.2. In Section IV-B, we detail our model architecture, which comprises multimodal feature fusion encoder in Section IV-B.1 and a transformer-based decoder in Section IV-B.2.

A. Modeling Navigation Information

1) Driving command: The current frame is defined as t. For driving commands $\mathbf{c}$ based on a fixed time horizon T [7],the vehicle's future trajectory is defined as $\tau =$

$\{(x_{t+1}, y_{t+1}), (x_{t+2}, y_{t+2}), \ldots, (x_{t+T}, y_{t+T})\}$ in the ego vehicle's coordinate system. Here (x_{t+n}, y_{t+n}) represents the lateral and longitudinal coordinates at time step $t+n$, respectively. The longitudinal coordinate y_{t+T} at the trajectory's endpoint represents the vehicle's longitudinal deviation relative to its position at time $t+T$. The driving command is then determined based on an offset threshold C, which defines the acceptable range of longitudinal deviation.

$$\mathbf{c} = \begin{cases} [1 \quad 0 \quad 0] & \text{if } y_t > C, \\ [0 \quad 1 \quad 0] & \text{if } |y_t| \leq C, \\ [0 \quad 0 \quad 1] & \text{if } y_t < -C. \end{cases} \tag{1}$$

where c is a one-hot vector with elements corresponding to Turn Left, Go Forward, and Turn Right. For driving commands based on a fixed path length D [10], τ is defined as the centerline of the predefined route over a specified distance D. In this case, the trajectory is parameterized by distance rather than time, and the longitudinal deviation is evaluated at the endpoint of the path. If no route is available or the route is too damaged to be reliably followed, the driving command will be set to none.

2) Sequential navigation guidance: In real-world driving scenarios, most driving behaviors are guided by specific navigation information, often facilitated by tools such as Google Maps [17]. Navigation information typically comprises two key components: a pre-planned global route R, generated using [36], and real-time turn-by-turn (TBT) information I. The global route, when transformed from the world coordinate system to the vehicle coordinate system, serves as a reference line for the vehicle's direction of driving. Simultaneously, TBT information, which includes high-level textual prompts, provides immediate guidance for local maneuvers. We construct SNG by integrating the navigation path and turn-by-turn (TBT) information, as illustrated in Fig. 3. Specifically, a predefined 40m route is selected as a reference, and sampling is performed to obtain the navigation path $P = \{(\hat{x}_1, \hat{y}_1), (\hat{x}_2, \hat{y}_2), \ldots, (\hat{x}_{N_P}, \hat{y}_{N_p})\}$, N_p represents the number of navigation points. The TBT information includes sequential action instructions, lane guidance, and traffic conditions. The vehicle's ego state $S_t = (v_x^t, v_y^t, a_x^t, a_y^t)$, where v_x and v_y denote the longitudinal and lateral velocities, and a_x and a_y represent the longitudinal and lateral accelerations at time, along with the front-view image and the predefined route, are provided as input prompts to the GPT-4o [31] model for generating the TBT information.

B. Network Architecture

We employ LLaVA [39] as the backbone, integrating a large language model, Qwen2.5 [40], and a vision encoder, SigLIP [41]. Following the method [42], we incorporate additional encoders specifically designed for the navigation path and ego state, thereby augmenting the model's capacity to process multimodal inputs. Our method can also integrate various existing perception modules used in previous end-to-end planners [7], [20]. By aligning the feature dimensions output by the perception module with the feature space of the LLM backbone, the model can seamlessly incorporate BEV (Bird's Eye View) features or LiDAR features. The hidden states output by the transformer backbone are enhanced through a cross-attention mechanism to improve interaction with the navigation context.

1) Scene representation: For a driving scenario, the input is represented as TBT information I, navigation path $P = \{(\hat{x}_1, \hat{y}_1), (\hat{x}_2, \hat{y}_2), \ldots, (\hat{x}_{N_P}, \hat{y}_{N_p})\}$, multi-view images $V = (M^1, M^2, ...M^{N_M})$, and the vehicle's ego state $S_t = (v_x^t, v_y^t, a_x^t, a_y^t)$. The TBT information I is encoded into a $F_T \in \mathbb{R}^{N_T \times H}$ through LLM tokenizer, where N_T denotes the number of text tokens and H corresponds to the feature dimension of the LLM's transformer backbone. Similarly, P is encoded into $F_P \in \mathbb{R}^{N_P \times H}$ through multi-layer perceptron (MLP) layers. We use a pre-trained SigLIP vision encoder [41] to extract features F'_M from multi-view images, which are then projected into $F_M \in \mathbb{R}^{N' \times H}$ via a linear transformation. To mitigate overfitting on ego state inputs [15], [16], we adopt an attention-based state dropout encoder (SDE) inspired by [43], which applies dropout to each state channel and processes ego state to $F_E \in \mathbb{R}^{4 \times H}$.

2) Transformer Decoder: After obtaining the representations of the driving scenario, all features are concatenated into F and fed as input to the transformer backbone.

$$F = \text{Concat}(F_T, F_P, F_M, F_E) \tag{2}$$

The hidden states output by the transformer backbone interact with the navigation query through a cross-attention module, followed by MLP layers to predict the trajectory. The loss function consists of the L1 loss between the predicted and the ground truth trajectory.

$$\mathcal{L} = \|\hat{\tau} - \tau\| \tag{3}$$

where the $\hat{\tau}$ denotes the predicted trajectory and τ denotes the future ground truth trajectory.

V. Experiments

We evaluate our method in Bench2Drive [18], a closed-loop evaluation benchmark under CARLA Leaderboard 2.0 [19] for end-to-end autonomous driving. The base set, consisting of 1,000 clips, is used for training, while the model is evaluated on 220 official routes. Additionally, we conduct closed-loop experiments in the NAVSIM benchmark [10] to assess its performance in real-world scenarios.

A. Implementation Details

We employ pre-trained Qwen2.5-0.5B as the transformer backbone and pre-trained SigLIP-So400M as the vision encoder, with a patch size of 14 and an image size of 384. In the state dropout encoder (SDE), we apply a dropout rate of 0.5 to the four ego state channels. The visual inputs consist of front and rear camera images, which undergo additional downsampling after passing through the vision encoder. The TBT information is only used in the NAVSIM experiment. We use a learning rate of 1e-6 with a cosine annealing schedule and a warmup ratio of 0.03. The model is trained

Method	Open-loop Metric	Closed-loop Metric			
	Avg. L2 ↓	Driving Score ↑	Success Rate (%) ↑	Efficiency ↑	Comfortness ↑
AD-MLP [15]	3.64	18.05	0.00	48.45	22.63
UniAD-Tiny [3]	0.80	40.73	13.18	123.92	**47.04**
UniAD-Base [3]	**0.73**	45.81	16.36	129.21	43.58
VAD [7]	0.91	42.35	15.00	157.94	46.01
Ours	0.82	**67.17**	**35.90**	**158.58**	22.30
TCP* [24]	1.70	40.70	15.00	54.26	47.80
TCP-ctrl* [24]	–	30.47	7.27	55.97	**51.51**
TCP-traj* [24]	1.70	59.90	30.00	76.54	18.08
TCP-traj w/o distillation [24]	1.96	49.30	20.45	**78.78**	22.96
ThinkTwice* [26]	0.95	62.44	31.23	69.33	16.22
DriveAdapter* [25]	1.01	**64.22**	**33.08**	70.22	16.01

TABLE I: **Open-loop and Closed-loop Results in Bench2Drive**. All results are trained on the base training set. Avg. L2 is averaged over the predictions in 2 seconds under 2Hz. * denotes expert feature distillation.

Method	Ability ↑					
	Merging	Overtaking	Emergency Brake	Give Way	Traffic Sign	Mean
AD-MLP [15]	0.00	0.00	0.00	0.00	4.35	0.87
UniAD-Tiny [3]	8.89	9.33	20.00	20.00	15.43	14.73
UniAD-Base [3]	14.10	17.78	21.67	10.00	14.21	15.55
VAD [7]	8.11	**24.44**	18.64	20.00	19.15	18.07
Ours	**33.75**	11.11	**46.60**	**50.00**	**50.00**	**38.08**
TCP* [24]	16.18	20.00	20.00	10.00	6.99	14.63
TCP-ctrl* [24]	10.29	4.44	10.00	10.00	6.45	8.23
TCP-traj* [24]	8.89	24.29	**51.67**	40.00	46.28	34.22
TCP-traj w/o distillation [24]	17.14	6.67	40.00	**50.00**	28.72	28.51
ThinkTwice* [26]	27.38	18.42	35.82	**50.00**	54.23	37.17
DriveAdapter* [25]	**28.82**	**26.38**	48.76	**50.00**	**56.43**	**42.08**

TABLE II: **Multi-Ability Results in Bench2Drive**. All results are trained on the base training set. * denotes expert feature distillation.

Method	NC↑	DAC↑	TTC↑	Comf.↑	EP↑	PDMS↑
UniAD [3]	97.8	91.9	92.9	**100**	78.8	83.4
PARA-Drive [11]	97.9	92.4	93.0	99.8	79.3	84.0
LTF [9]	97.4	92.8	92.4	**100**	79.0	83.8
Transfuser [9]	97.7	92.8	92.8	**100**	79.2	84.0
Transfuser†	97.9	93.8	93.5	**100**	79.6	85.1
DRAMA [37]	98.0	93.1	**94.8**	**100**	80.1	85.5
Hydra-MDP [38]	**98.3**	96.0	94.6	**100**	78.7	86.5
Ours	97.7	**97.1**	93.1	**100**	**83.1**	**88.2**

TABLE III: **Comparison on NAVSIM navtest split with closed-loop metrics**. † represents the results we reproduced. PDM score (PDMS) [10] is weighted aggregation of several sub-scores: no at-fault collisions (NC), drivable area compliance (DAC), time-to-collision (TTC), comfort (Comf.), and ego progress (EP).

on 8 × NVIDIA A100 GPU 80G with a per-GPU batch size of 8 for 10 epochs.

B. Main results

We compare our method with other E2E-AD methods in both Bench2Drive and NAVSIM. Table I and Table II present the results in Bench2Drive, showing that our method achieves state-of-the-art performance. Due to the lack of accurate modeling of navigation information, models based on driving commands are prone to losing their targets during the planning process, resulting in trajectories that tend to deviate in random directions. Furthermore, the presence of cumulative errors in closed-loop experiments further degrades the performance of driving command-based methods, leading to poor performance in metrics such as task completion rate and driving score. While, our SNG based model has significantly outperformed existing methods in these metrics. Compared to UniAD-Base [3], our method surpasses it by 46.6% and 119.4% in terms of Driving Score and Success Rate, respectively. In the mean Multi-Ability score, our method outperforms VAD [7] by 110.7%. Our

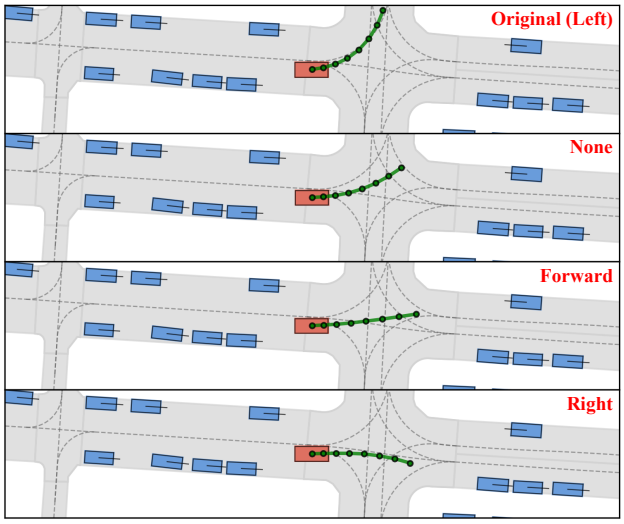

Fig. 4: We demonstrate the impact of introducing noise to the driving command on the predicted trajectory during the inference process of the Transfuser [9]. The experiments are based on the correctly trained checkpoint of the Original model in Table IV. The scenario involved an open intersection without traffic lights.

method has also demonstrated superior performance across various sub-ability tests, except for overtaking tasks. This limitation stems from the use of only front and rear images in the Bench2Drive experiments, which results in the absence of lateral perspective information and consequently leads to failures in certain overtaking scenarios. However, since our pipeline can seamlessly integrate existing perception modules, this limitation can be readily addressed in future work. Furthermore, in comparison with the expert feature distillation approach, our method continues to maintain a leading position in both Driving Score and Success Rate.

Table III presents the results on NAVSIM [10]. The performance of Hydra-MDP [38] is enhanced through additional training to optimize for the EP evaluation metric, utilizing supplementary supervision and weighted confidence post-processing. Despite this, our model achieves SOTA performance without relying on any supervision from perception tasks. Notably, our model exhibits improvement on the DAC (drivable area compliance) metric, which underscores the enhanced efficacy of the proposed SNG.

C. Ablation study

1) Driving Command Fails to model Navigation information: Undoubtedly, clear navigation information plays an important role in autonomous driving systems, providing essential guidance for determining the vehicle's direction of driving. However, as shown in Table IV, after introducing varying levels of noise into the driving command, the model can yield results that are comparable to, or surpass the official results on metrics on PDM score and others. The model demonstrates performance comparable to the original under random and left command. It exhibits a slight decline in performance when right or forward commands are utilized. Notably, the model achieves enhanced performance in the absence of any command. Regarding the NC and DAC metrics, which are more sensitive to navigation information, the introduction of errors in the command shows no significant changes in these metrics.

Command	NC↑	DAC↑	TTC↑	Comf.↑	EP↑	PDMS↑
Original	97.84	93.77	93.53	99.98	79.58	85.14
None	97.74	94.55	93.13	99.98	80.01	85.51
Random	97.67	94.04	93.19	99.97	79.67	85.11
Left	97.83	94.17	93.32	99.98	79.61	85.22
Right	97.69	93.46	93.15	99.99	78.88	84.44
Forward	97.66	93.15	93.08	99.99	78.89	84.23

TABLE IV: **Command ablation results in NAVSIM**. We conducted experiments on NAVSIM [10] using Transfuser [9] by modifying the input driving commands. Original: ground truth driving command; None: no driving command added; Random: random driving command; Left, Right, Forward: fixed driving commands. All results are obtained under identical training parameters. The performance of our reproduced Original surpasses the official results reported in [10]. The introduction of noise into the driving command has minimal impact on the final planning performance.

ID	Navigation Path	TBT Information	Driving Command	NC↑	DAC↑	PDMS↑
0	–	–	–	97.2	95.1	85.9
1	–	–	✓	97.5	95.3	86.1
2	–	✓	–	97.6	95.2	86.4
3	2×20	–	–	97.8	95.1	86.4
4	2×20	✓	–	97.5	96.1	87.6
5	4×10	–	–	97.5	96.6	87.7
6	4×10	✓	–	97.7	97.1	88.2
7	8×5	–	–	97.5	96.2	87.2
8	8×5	✓	–	97.5	96.6	87.6

TABLE V: **Ablation of navigation information representation.** We conduct ablation studies on the sampling interval of the navigation path, TBT information and driving commands.

We present qualitative results in Fig. 4, based on an open intersection scenario without traffic lights, which allows the planner to choose any direction. However, when driving commands such as "Go Forward" or "Turn Right" are applied, the model fails to generate trajectories aligned with the intended direction. Particularly, in the absence of driving command inputs, the model produces trajectories that extend beyond the drivable area.

2) Driving command vs. Sequential navigation guidance: As illustrated in Table V (ID 0-4), the results obtained without any navigation information (ID 0) and with driving commands (ID 1) exhibit no significant difference. The performance achieved by solely utilizing TBT information (ID 2) surpasses that of both (ID 0) and (ID 1). Notably, when employing only two points spaced 20 meters apart as the navigation path (ID 3), the model's performance is comparable to that of (ID 2), indicating that the navigation path can provide the model with effective spatial reference under

Fig. 5: Hardware platform.

sparse sampling. The model achieves its optimal performance when both the navigation path and TBT information are used as sequential navigation guidance (ID 4). These results demonstrate that sequential navigation guidance models navigation information more effectively than driving commands alone. Specifically, the navigation path provides long-term trajectory constraints, while TBT information offers real-time decision-making logic, such as road traffic conditions. The synergy between these two elements mitigates the limitations associated with relying on a single modality.

3) Optimal combination of navigation path & TBT information: We further investigated the optimal combination of navigation paths and TBT information, as practical operations often face challenges in consistently obtaining sufficiently dense navigation paths or accurate TBT information. As shown in Table V (ID 3-8), we systematically evaluated the impact of varying densities of navigation path points and the inclusion of TBT information on the results. Comparing (ID 3, 5, 7), it's shown that 4 navigation points spaced 10 meters apart yielded the best performance. Both excessively sparse and overly dense configurations led to diminished performance. Sparse navigation points fail to accurately model the reference path, while overly dense points impose excessive constraints on the model, resulting in poor performance in scenarios such as obstacle avoidance. Across (ID 3-8), the inclusion of TBT information consistently improved performance under varying navigation point densities. In conclusion, a moderate-density navigation path combined with TBT information serves as an effective SNG setting, optimally modeling navigation information and maximizing the model's planning performance.

D. Real world experiments

To further evaluate our proposed sequential navigation guidance and model in real-world scenarios, we established a validation platform for physical vehicles, as illustrated in Fig. 5. The platform is equipped with a primary LiDAR, the Innovusion Falcon 300, and a surround-view camera system comprising five AR0820 cameras with a 120-degree horizontal field of view (HFOV) and two AR0820 cameras with a 70-degree HFOV. The onboard computing unit consists of dual Orin modules. We collect and create in-house dataset and conduct experiments. As illustrated in Fig. 6, our method demonstrates robust performance in scenarios involving both straight paths and turns. Notably, as shown in Fig. 6 (c), even under conditions of road construction on the main route, our method successfully plans the correct left-turn trajectory.

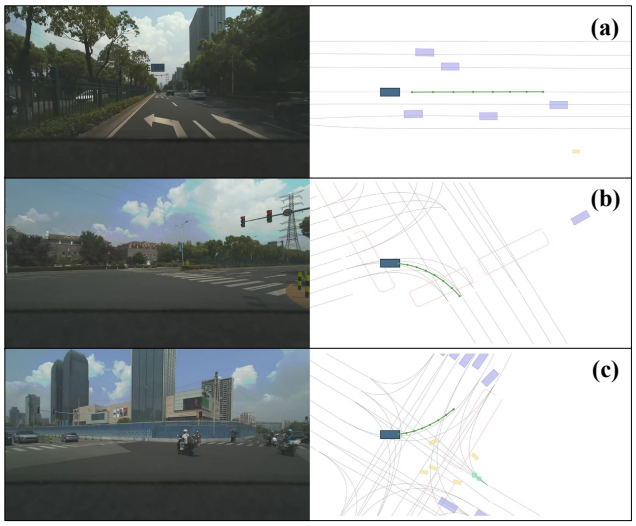

Fig. 6: Qualitative analysis of real-world scenarios. (a) depicts a straight-driving scenario. (b) represent right-turn scenarios. (c) illustrates a left-turn scenario with construction on the route.

VI. Conclusions

In this study, we investigate the limitations of current end-to-end autonomous driving systems in utilizing navigation information and introduce a novel representation of navigation information, termed Sequential Navigation Guidance, which integrates long-term trajectory constraints and real-time decision logic. Our model, based on SNG, achieves superior performance in closed-loop evaluations without the need for supervision from perception tasks. Experiments conducted in real-world scenarios further confirm the robustness and practical applicability of our approach.

References

[1] T. Wu, A. Luo, R. Huang, H. Cheng, and Y. Zhao, "End-to-end driving model for steering control of autonomous vehicles with future spatiotemporal features," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 950–955.

[2] Y. Xiao, F. Codevilla, A. Gurram, O. Urfalioglu, and A. M. López, "Multimodal end-to-end autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 1, pp. 537–547, 2022.

[3] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 853–17 862.

[4] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, and H. Li, "End-to-end autonomous driving: Challenges and frontiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 10 164–10 183, 2024.

[5] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, "Towards fully autonomous driving: Systems and algorithms," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 163–168.

[6] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 2090–2096.

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

[8] S. Hu, L. Chen, P. Wu, H. Li, J. Yan, and D. Tao, "St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning," in *European Conference on Computer Vision*. Springer, 2022, pp. 533–549.

[9] K. Chitta, A. Prakash, B. Jaeger, Z. Yu, K. Renz, and A. Geiger, "Transfuser: Imitation with transformer-based sensor fusion for autonomous driving," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 11, pp. 12 878–12 895, 2023.

[10] D. Dauner, M. Hallgarten, T. Li, X. Weng, Z. Huang, Z. Yang, H. Li, I. Gilitschenski, B. Ivanovic, M. Pavone, A. Geiger, and K. Chitta, "Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[11] X. Weng, B. Ivanovic, Y. Wang, Y. Wang, and M. Pavone, "Paradrive: Parallelized architecture for real-time autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 15 449–15 458.

[12] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024, pp. 256–274.

[13] W. Zheng, R. Song, X. Guo, C. Zhang, and L. Chen, "Genad: Generative end-to-end autonomous driving," in *Proceedings of the European Conference on Computer Vision (ECCV)*, A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, and G. Varol, Eds., 2024, pp. 87–104.

[14] W. Xu, J. Pan, J. Wei, and J. M. Dolan, "Motion planning under uncertainty for on-road autonomous driving," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 2507–2512.

[15] J.-T. Zhai, Z. Feng, J. Du, Y. Mao, J.-J. Liu, Z. Tan, Y. Zhang, X. Ye, and J. Wang, "Rethinking the open-loop evaluation of end-to-end autonomous driving in nuscenes," *arXiv preprint arXiv:2305.10430*, 2023.

[16] 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.

[17] G. Svennerberg, *Beginning google maps API 3*. Apress, 2010.

[18] X. Jia, Z. Yang, Q. Li, Z. Zhang, and J. Yan, "Bench2drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving," in *NeurIPS 2024 Datasets and Benchmarks Track*, 2024.

[19] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[20] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Q. Yu, and J. Dai, "Bevformer: learning bird's-eye-view representation from lidar-camera via spatiotemporal transformers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[21] L. Zhang, P. Li, J. Chen, and S. Shen, "Trajectory prediction with graph-based dual-scale context fusion," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 11 374–11 381.

[22] M. Naumann and C. Stiller, "Aib-mdp: Continuous probabilistic motion planning for automated vehicles by leveraging action independent belief spaces," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 6373–6380.

[23] S. Casas, A. Sadat, and R. Urtasun, "Mp3: A unified model to map, perceive, predict and plan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 403–14 412.

[24] P. Wu, X. Jia, L. Chen, J. Yan, H. Li, and Y. Qiao, "Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline," *Advances in Neural Information Processing Systems*, vol. 35, pp. 6119–6132, 2022.

[25] X. Jia, Y. Gao, L. Chen, J. Yan, P. L. Liu, and H. Li, "Driveadapter: Breaking the coupling barrier of perception and planning in end-to-end autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 7953–7963.

[26] X. Jia, P. Wu, L. Chen, J. Xie, C. He, J. Yan, and H. Li, "Think twice before driving: Towards scalable decoders for end-to-end autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 21 983–21 994.

[27] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[28] K. T. e. a. H. Caesar, J. Kabzan, "Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles," in *CVPR ADP3 workshop*, 2021.

[29] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454.

[30] Q. Li, Z. Peng, L. Feng, Q. Zhang, Z. Xue, and B. Zhou, "Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 45, no. 3, pp. 3461–3475, 2022.

[31] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[32] G. Team, P. Georgiev, V. I. Lei, R. Burnell, L. Bai, A. Gulati, G. Tanzer, D. Vincent, Z. Pan, S. Wang *et al.*, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," *arXiv preprint arXiv:2403.05530*, 2024.

[33] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

[34] Z. Xu, Y. Zhang, E. Xie, Z. Zhao, Y. Guo, K.-Y. K. Wong, Z. Li, and H. Zhao, "Drivegpt4: Interpretable end-to-end autonomous driving via large language model," *IEEE Robotics and Automation Letters*, vol. 9, no. 10, pp. 8186–8193, 2024.

[35] H. Shao, Y. Hu, L. Wang, G. Song, S. L. Waslander, Y. Liu, and H. Li, "Lmdrive: Closed-loop end-to-end driving with large language models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 15 120–15 130.

[36] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[37] 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.

[38] 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.

[39] B. Li, Y. Zhang, D. Guo, R. Zhang, F. Li, H. Zhang, K. Zhang, P. Zhang, Y. Li, Z. Liu *et al.*, "Llava-onevision: Easy visual task transfer," *arXiv preprint arXiv:2408.03326*, 2024.

[40] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei *et al.*, "Qwen2. 5 technical report," *arXiv preprint arXiv:2412.15115*, 2024.

[41] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, "Sigmoid loss for language image pre-training," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 11 975–11 986.

[42] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," in *International conference on machine learning*. PMLR, 2023, pp. 19 730–19 742.

[43] J. Cheng, Y. Chen, X. Mei, B. Yang, B. Li, and M. Liu, "Rethinking imitation-based planners for autonomous driving," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 14 123–14 130.