A Multi-Modality End-to-End Autonomous Driving System

Bryan Pineda
Stanford University
450 Jane Stanford Way, Stanford, CA 94405
bamp86@stanford.edu

Ziyu Li
Stanford University
450 Jane Stanford Way, Stanford, CA 94405
ziyu425@stanford.edu

Kamal Mohammed ElMallah
Stanford University
450 Jane Stanford Way, Stanford, CA 94405
kamale@stanford.edu

Abstract

Recent developments in end-to-end learning have shown promising results for autonomous driving by replacing traditional modular pipelines with unified frameworks. However, many existing approaches are restricted to simulated environments or lack real-world robustness. In this work, we extend the TransFuser architecture to operate on real-world, multi-modal inputs by fusing panoramic RGB images, LiDAR-based bird's-eye view (BEV) features, and ego-velocity measurements. Our model autoregressively predicts a sequence of future waypoints over a 4-second horizon and incorporates semantic BEV supervision to enhance spatial understanding. We train and evaluate our approach on the nuScenes dataset, demonstrating strong performance in diverse urban driving scenarios.

1. Introduction

Recent advances in GPUs and computational power have significantly accelerated the development of data-driven autonomous driving systems. Traditional pipeline-based frameworks decompose the task into perception, prediction, and planning modules, offering interpretability and modularity. However, end-to-end (E2E) approaches, which directly map sensor inputs to control actions, are gaining attention for their simplicity, efficiency, and potential performance gains.

Autonomous driving systems operate in complex and dynamic real-world environments, where relying on a single sensor modality is often insufficient for robust and reliable perception, prediction, and planning. Leveraging multimodal data—such as camera images, LiDAR point clouds, radar signals, GPS, and inertial measurements—has become essential for enhancing situational awareness and ensuring safety. Each modality provides unique and complementary information: cameras capture rich semantic context, LiDAR delivers precise 3D spatial structure, and radar performs well in adverse weather conditions. By fusing these diverse sources of information, multimodal systems can achieve greater redundancy, accuracy, and resilience to sensor failure or environmental noise.

TRANSFUSER [11] introduces a multi-modal fusion Transformer that integrates global context and cross-modal interactions during feature extraction from LiDAR and camera images. It is designed for end-to-end autonomous driving via an auto-regressive waypoint prediction framework. TransFuser achieves a driving score of 47.3 on the CARLA Longest6 Benchmark [5]. While its design is computationally efficient and more suitable for industrial deployment, its training and evaluation are limited to simulated environments.

In this study, we propose to build our framework upon TRANSFUSER [11], adapting it to account for a real world multi-modality data, camera and LiDAR point clouds. Unlike the original TransFuser, which utilizes LiDAR and monocular images from a simultaneous environment, our method takes a single panoramic image and processed LiDAR BEV features as input, augmented with the current ego-velocity. Temporal reasoning is handled by an autoregressive decoder that predicts a sequence of future waypoints over a 4-second horizon. To ensure real-world applicability and robustness, we train and evaluate our model on the nuScenes dataset [2], which provides diverse, real-world urban driving scenarios.

Contributions. Our key contributions are:

- We extend the TransFuser architecture to handle real-world panoramic RGB and LiDAR BEV data from nuScenes, with temporal reasoning enabled through autoregressive waypoint prediction.

- We incorporate semantic BEV supervision to enhance LiDAR representation learning.
- We provide a thorough evaluation with both quantitative metrics and qualitative trajectory visualizations across urban scenes.

2. Related Work

The rise of GPUs and the steady advancement in computational resources have greatly empowered data-driven learning methods, leading to remarkable breakthroughs in the field of autonomous driving[6]. Many research have been done to improve the performance of each module in a pipeline framework, including perception[8, 14], prediction[7] and planning[13], in the autonomous driving system. The pipeline framework has an advantage of interpretation enabling people can understand what the output of each component and avoid unexpected behavior[12] from the final output which is very important of autonomous driving safety.

Another framework for autonomous driving is the end-to-end system, which takes raw sensor data as input and directly outputs planning decisions or control actions. In this approach, features are forward-propagated through sequential modules, and the final planning output is optimized via backpropagation across all modules. This architecture offers advantages such as simplicity, potential for improved performance, and computational efficiency[3].

Since the introduction of ALVINN [9], which employed a three-layer neural network for autonomous land vehicle navigation, numerous studies have explored end-to-end autonomous driving using machine learning techniques. A convolutional neural network (CNN)[1] was proposed that takes raw camera images as input and directly outputs low-level steering commands. During training, images from left, right, and center cameras were used for data augmentation, while during inference, only the center camera was used to predict the steering angle. Visualization of the internal feature maps revealed that the network was able to learn salient information, such as road boundaries. In real-world tests, the system successfully drove a car autonomously for up to 10 kilometers on a highway without human intervention.

Behavior cloning, an off-policy method where a network is trained using offline data and evaluated online, often suffers from the compounding error problem due to covariate shift. To address this, Dataset aggregation was combined dataset aggregation, critical state sampling, and replay buffer mechanisms into an on-policy learning framework[10]. They trained a ResNet-34-based model using RGB images, ego speed, and high-level navigation commands as input, and predicted low-level control signals (steering, throttle, brake) as output [4]. Their approach demonstrated improved performance under previously unseen towns and weather conditions.

Beyond CNN-based architectures, transformers have also been adopted for autonomous driving since 2021. Transfuser[11] was proposed, a transformer-based framework that integrates multi-modal data (images and LiDAR) for imitation learning. TransFuser achieved state-of-the-art performance in terms of driving score on the CARLA benchmark. More recently, several works have explored the use of large language models (LLMs) for autonomous driving. DriveGPT [15] leverages a multimodal LLM to enhance the interpretability of end-to-end driving systems. In addition to predicting control commands (speed and steering angle), DriveGPT also provides natural language explanations of driving behavior. It uses CLIP to extract features from video segments, which are then tokenized and fed into an LLM for both control prediction and textual reasoning.

3. Dataset Overview

We use the nuScenes dataset [2], a large-scale autonomous driving benchmark collected in Boston and Singapore. It contains 1,000 urban driving scenes, each 20 seconds long and sampled at 2 Hz, totaling 1.4 million images. Scenes include annotations for 23 object classes and support tasks such as detection, tracking, and planning. The dataset features varied traffic density, weather, and lighting conditions.

3.1. Sensor Suite and Data Modalities

Each scene includes six surround-view RGB cameras (front, front-left, front-right, back-left, back-right, back), a 32-beam LiDAR, five radars, and high-definition maps. Cameras operate at 12 Hz, LiDAR at 20 Hz, and radar at 13 Hz. Vehicle pose, velocity, and acceleration are also provided for motion compensation and forecasting.

3.2. Annotations

nuScenes provides 3D bounding boxes for tracked agents (vehicles, pedestrians, cyclists) at 2 Hz, with position, orientation, and attributes. It also includes semantic map elements such as lanes, road boundaries, and crosswalks, enabling map-aware modeling.

4. Preprocessing Pipeline

4.1. Camera Input

We utilize three forward-facing camera views: front-left, front, and front-right. Each image has a spatial resolution of 1600×900 pixels with 3 color channels, resulting in a shape of 3×900×1600. To minimize potential boundary artifacts arising from concatenation, we crop 100 pixels from both

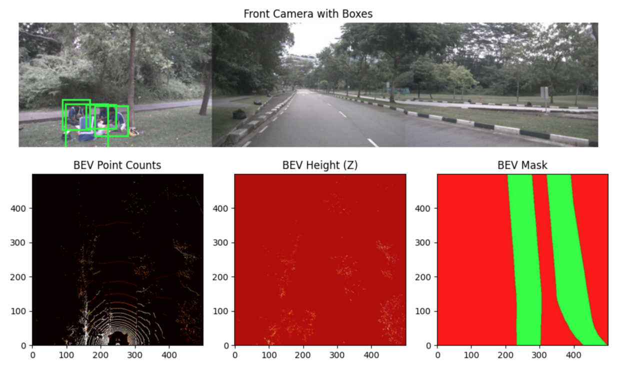

Figure 1. Visualization of camera (top) and BEV feature map (bottom).

the left and right borders of each image, yielding a post-crop resolution of 3×900×1400. The cropped images are then concatenated along the width dimension to produce a single panoramic input tensor of shape 3×900×4200. This composite view captures a wide forward field-of-view and serves as the input to the image encoder module.

4.2. LiDAR Input

To construct a bird's-eye view (BEV) representation from the LiDAR point cloud, we discretize a 50×50 meter region around the ego vehicle into a 500×500 grid with a spatial resolution of 0.1 meters per pixel. The ego vehicle is centered at the bottom middle of the BEV frame. Two feature channels are encoded per pixel:

- **Point density**: The number of LiDAR points falling into each pixel.
- **Maximum height**: The maximum height of the points within the pixel.

To suppress outliers and reduce ground clutter, we retain only points with z-values in the range $[-1\,\text{m}, 5\,\text{m}]$. The resulting tensor has shape 2×500×500 and serves as input to the LiDAR encoder.

4.3. BEV Semantic Mask

In addition to LiDAR features, we generate a semantic segmentation mask in the BEV frame. This mask is derived from nuScenes annotations and classifies each pixel into one of three categories: background, road surface, and annotated vehicles. The semantic mask is used as supervision during training to evaluate the LiDAR encoder's ability to capture semantic scene structure from raw point clouds.

5. Method

We build on the open-source implementation of TransFuser [11], a Transformer-based sensor fusion model originally designed for end-to-end driving in the CARLA simulator. Our modifications adapt it to handle real-world, multi-frame data from nuScenes and extend it with semantic supervision and temporal autoregressive decoding. Below, we describe the model inputs, outputs, and loss formulation, along with the learning algorithms used in each module.

5.1. Input and Output Formulation

Let $\mathbf{I}_t \in \mathbb{R}^{3\times H\times W}$ denote the panoramic RGB image at time t, and let $\mathbf{L}_t \in \mathbb{R}^{C_L\times H'\times W'}$ represent the BEV LiDAR input, where $C_L = 3$ (point density, and maximum height channels). In addition, the model receives a target navigation point $\mathbf{p} \in \mathbb{R}^2$ which is the groundtruth ego way point after 4 seconds and scalar ego velocity $v_t \in \mathbb{R}$. The ground truth BEV semantic mask is denoted as $\mathbf{Y}_{\text{bev}} \in \{0, 1, 2\}^{H'\times W'}$.

The model predicts:

- A sequence of future waypoints $\widehat{\mathbf{Y}}_{\text{wp}} = \{\widehat{\mathbf{y}}_1, \ldots, \widehat{\mathbf{y}}_T\} \in \mathbb{R}^{T\times 2}$.
- A semantic segmentation map $\widehat{\mathbf{Y}}_{\text{bev}} \in \mathbb{R}^{3\times H'\times W'}$ of the BEV.

5.2. Model Architecture

The model can be divided into 3 parts, a Transfuser Backbone, a waypoint prediction, and a BEV feature decoder part.

A Transfuser-based multimodal backbone (Transfuser-Backbone) fuses RGB images, BEV LiDAR maps, and fuse the features at each encoder layer. It employs two parallel CNN encoders, ImageEncoder and LidarEncoder, both adapted from the TIMM model library (both use RegNet32 in our case). The ImageEncoder normalizes image input $\mathbf{I}_t$ using ImageNet statistics and removes non-essential components like the classification head, retaining only the convolutional backbone and global average pooling to extract spatial features. Similarly, LidarEncoder modifies the first convolutional layer to accept a configurable number of input channels, while disabling classification-related components. At each layer $\mathbf{l}$, both encoders output high-dimensional spatial feature maps that are subsequently downsampled using adaptive average pooling into fixed grid resolutions $\mathbf{f}^{\mathbf{l}}_{\text{rgb}} \in \mathbb{R}^{d\times N}$ and $\mathbf{f}^{\mathbf{l}}_{\text{lidar}} \in \mathbb{R}^{d\times M}$ (5×22 for image, 8×8 for LiDAR). These features are then flattened and concatenated to form a unified token sequence, which is passed into a two-stage transformer-based fusion module. We compute the fused feature as:

$$\mathbf{f}^{\mathbf{l}}_{\text{fused}} = \text{Transformer}([\mathbf{f}^{\mathbf{l}}_{\text{rgb}}; \mathbf{f}^{\mathbf{l}}_{\text{lidar}}]), \tag{1}$$

This fused output $\mathbf{f}^{\mathbf{l}}_{\text{fused}}$ is then split back into two modality-specific components: an updated image feature $\mathbf{f}^{\mathbf{l}}_{\text{fused image}}$ and an updated LiDAR feature$\mathbf{f}^{\mathbf{l}}_{\text{fused lidar}}$. Each is added element-wise to the corresponding original encoder feature at that layer, and the result is fed forward as the input to the next encoder stage in both branches.

For the ego way point prediction, the pooled fused feature vector $\mathbf{z} \in \mathbb{R}^d$ of the transfuser backbone is passed through a fully connected layer and used as the initial hidden state of a GRU decoder. The GRU autoregressively generates 2D displacements, conditioned on the previous output and the target point. At each step t, the decoder computes:

$$\mathbf{h}_t = \mathrm{GRU}([\mathbf{x}_t; \mathbf{p}], \mathbf{h}_{t-1}) \tag{2}$$

$$\Delta\widehat{\mathbf{y}}_t = \mathrm{MLP}(\mathbf{h}_t) \tag{3}$$

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \Delta\widehat{\mathbf{y}}_t \tag{4}$$

where $\mathbf{x}_t \in \mathbb{R}^2$ is the current predicted position, and $\mathbf{h}_t$ is the GRU hidden state. The final output is the sequence $\{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$.

For the bev decoder part, the intermediate feature map from the LiDAR encoder is decoded into a BEV semantic mask using a simple convolutional head:

$$\widehat{\mathbf{Y}}_{\text{bev}} = \mathrm{Upsample}(\mathrm{ConvBlock}(\mathbf{f}_{\text{bev}})). \tag{5}$$

This output is supervised with pixel-wise cross-entropy loss using the annotated BEV mask.

5.3. Loss Functions

The overall training objective combines two losses:

$$\mathcal{L} = \lambda_{\text{wp}} \cdot \mathcal{L}_{\text{wp}} + \lambda_{\text{bev}} \cdot \mathcal{L}_{\text{bev}}, \tag{6}$$

where we set $\lambda_{\text{wp}} = \lambda_{\text{bev}} = 1.0$ in our experiments.

We minimize the L1 distance between predicted and ground-truth waypoints:

$$\mathcal{L}_{\text{wp}} = \frac{1}{T}\sum_{t=1}^{T} \|\widehat{\mathbf{y}}_t - \mathbf{y}_t\|_1 \tag{7}$$

We use cross-entropy loss with class weights to address imbalance:

$$\mathcal{L}_{\text{bev}} = \mathrm{CE}(\widehat{\mathbf{Y}}_{\text{bev}}, \mathbf{Y}_{\text{bev}}) \tag{8}$$

5.4. Implementation Details

We adapt and extend the open-source TransFuser codebase, originally designed for simulation environments. Our contributions include:

- A custom data pipeline for loading panoramic images, LiDAR BEV tensors, and semantic labels from nuScenes.
- An autoregressive GRU-based decoder for future waypoint prediction.
- A semantic segmentation head supervising the LiDAR encoder via a BEV mask.
- Modifications to support autoregressive output decoding and ego-velocity conditioning.

Our implementation is built in PyTorch and optimized using AdamW with a learning rate of 10^{-4} and batch size of 16. We freeze the pretrained image and LiDAR encoders during training and fine-tune only the fusion and decoding modules.

6. Results and Discussion

We evaluated our model on 100 test samples across two distinct scenes and analyzed the waypoint prediction error over a 4-second horizon (8 sequential waypoints) in both the x and y directions, where the y-axis represents the along-track (longitudinal) motion and the x-axis corresponds to the cross-track (lateral) deviation.

As shown in Figure 2, the longitudinal (y) error increases until around 2 seconds (step 3–4) and then gradually decreases. This decline may be partially attributed to the model's use of the final target point as an input, which provides strong guidance near the endpoint but less constraint in the mid-horizon. As a result, the model finds it more difficult to predict intermediate waypoints along the trajectory, leading to greater error and variance in the middle steps. In contrast, the lateral (x) error increases steadily over time, with growing standard deviation, indicating that cross-track deviations are more difficult to predict at longer horizons due to accumulated uncertainty and maneuver variability. These trends highlight the importance of modeling temporal consistency and intent across the full trajectory, and suggest that incorporating intermediate planning cues or uncertainty-aware methods could improve prediction stability throughout the entire horizon.

Figures 5 and 6 illustrate qualitative results of our model's performance across different driving scenarios using front camera views and BEV representations. As shown in Figure 5, the predicted trajectories align closely with the ground truth when the vehicle is driving straight or when there is a single-lane road. In these simpler settings, the model demonstrates strong predictive accuracy and stability, likely due to reduced ambiguity in spatial planning and fewer alternative maneuver options. In contrast, Figure 6 presents two failure cases where the predicted trajectories deviate significantly from the ground truth, particularly during left or right turns in multi-lane environments. The increased prediction error in these scenarios may be attributed to the lack of fine-grained lane-level information in the BEV input. Specifically, while our model leverages road segmentation in the LiDAR encoder, it does not explicitly incorporate lane boundary segmentation, which is critical for precise lateral planning in complex intersections or merging

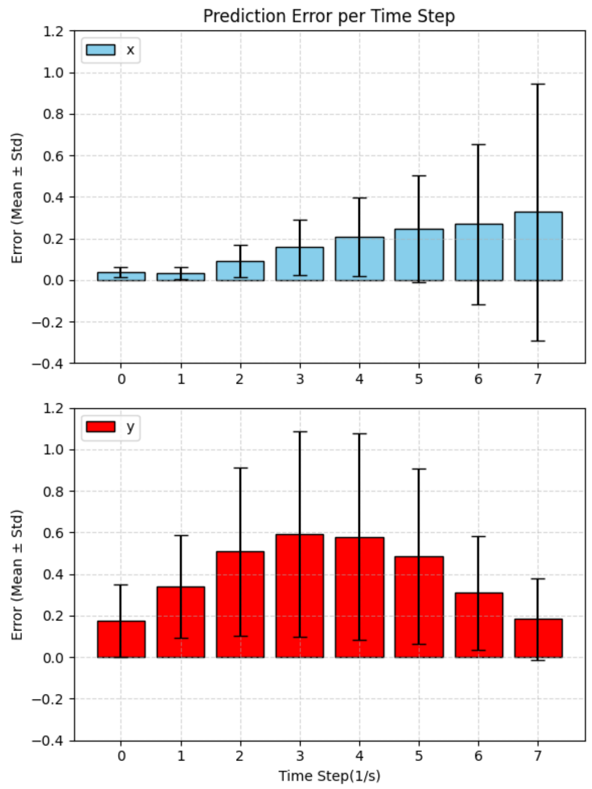

Figure 2. Error in x and y direction at different steps for 100 test samples.

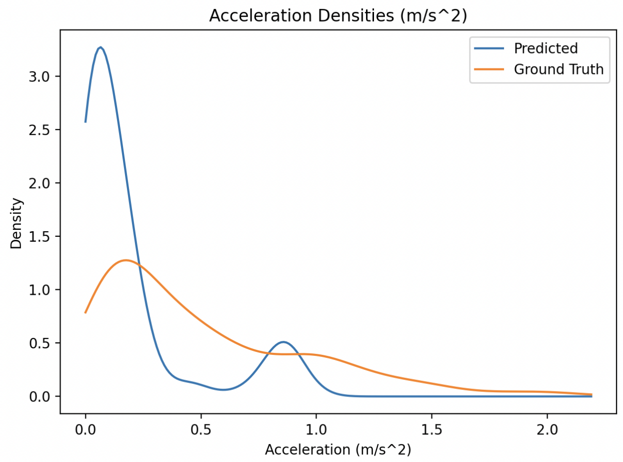

Figure 3. Acceleration Distribution for predicted waypoints and ground truth waypoints

scenarios. These observations suggest that enhancing the BEV representation with detailed lane semantics could substantially improve trajectory prediction performance, especially in scenes with high maneuver variability.

Additionally, we evaluated waypoint predictions kinematically, considering acceleration magnitude (Figure 3) and rotation magnitude (Figure 4). Rotationally, we find that the model is very close to our ground truth, demonstrating a lack of unnecessary steering. Although the model produces a lower turning angle median and average than the ground truth, signifying that in cases where extreme turns are necessary, the model may underperform in these cases. The acceleration is a more dramatic example of this observed undercompensation. The model produces waypoints that lack as high acceleration as the ground truth, generally, which can be both a positive and a negative aspect. The model generally provides a more efficient drive than the ground truth. However, in cases where extreme acceleration or deceleration is needed, the model may fail to do so fully. However, the Nuscience data set used does not contain many of these cases, as figure 3 demonstrates that the highest acceleration in the ground truth data sampled here was not larger than $2.19\ \mathrm{m/s^2}$. Because there are no extreme data points in which rapid acceleration or rotation are needed and human driving is not efficient, it is expected for the model to adopt a more conservative waypoint output in both acceleration and rotation, while being able to maintain low loss.

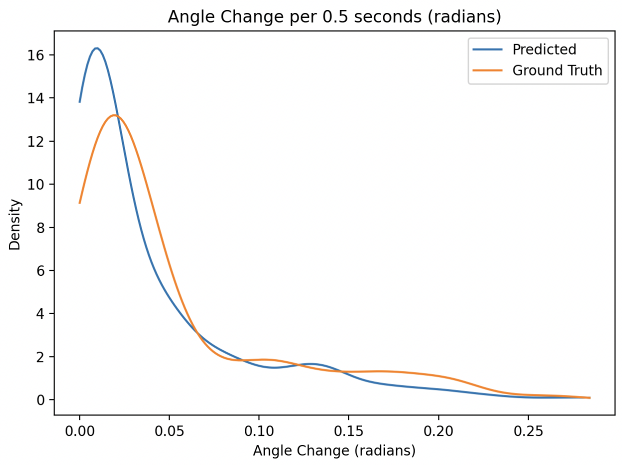

Figure 4. Change in angle of direction for predicted waypoints and ground truth waypoints over 0.5 seconds

7. Conclusion and Future Work

In this work, we present a real-world extension of the TransFuser framework for end-to-end autonomous driving by integrating panoramic RGB images, LiDAR-derived BEV features, and ego-velocity into an autoregressive waypoint prediction framework from the nuScenes dataset. Our model fuses multimodal sensor inputs using a Transformer-based architecture and autoregressively predicts future vehicle waypoints over a 4-second horizon. To enhance spatial understanding, we introduce semantic supervision via BEV segmentation, encouraging the model to learn meaningful geometric priors from raw LiDAR data.

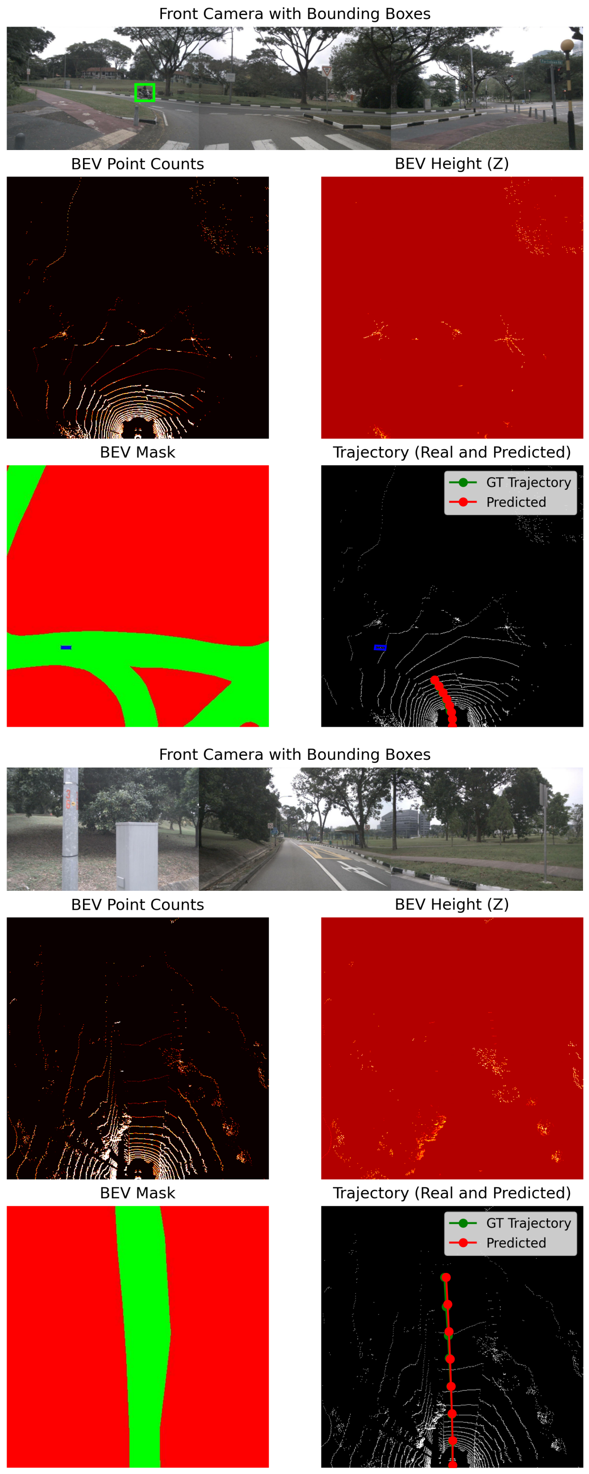

Figure 5. Visualization of camera (top) and BEV feature map (bottom) of 2 good prediction result. The lower right figures show the predicted trajectories in red lines and ground truth (GT) trajectories in green line.

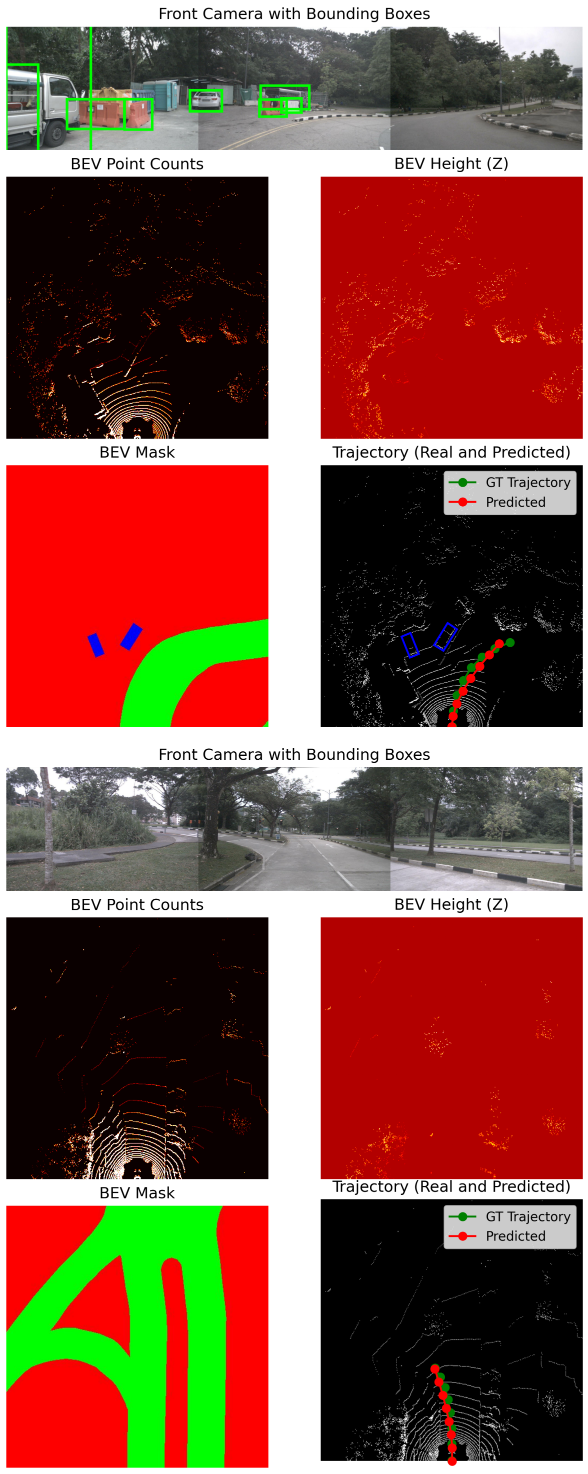

Figure 6. Visualization of camera (top) and BEV feature map (bottom) of 2 relative poor prediction result.

Quantitative results demonstrate that our model maintains low prediction error in straightforward driving scenarios, while qualitative analyses reveal its strengths and limitations in more complex, multi-lane environments. These findings highlight the value of combining semantic spatial cues with temporal fusion for trajectory forecasting. Furthermore, our adaptation of TransFuser to real-world data bridges the gap between simulation and deployment, paving the way for practical applications of multimodal imitation learning.

In future work, we aim to extend our architecture by incorporating a graph-based module that explicitly models interactions between the ego vehicle and surrounding agents. Additionally, we plan to improve temporal modeling by incorporating longer input sequences and memory-aware mechanisms to better capture motion intent and behavior over time. Finally, we intend to scale our training and evaluation across larger and more diverse datasets such as the Waymo Open Dataset, enabling broader generalization and robustness across complex driving scenarios. In addition to model architecture, we plan to train on more extreme driving scenarios using datasets provided by Waymo in order to explore how well the model performs in uncommon variable scenarios.

8. Contributions and Acknowledgements

This work builds upon the open-source implementation of Transfuser provided by the Autonomous Vision Group [https://github.com/autonomousvision/transfuser]. We sincerely thank the authors for releasing their codebase, which served as the foundation for our model development.

Ziyu Li conducted the related work section, processed parts of the nuScenes dataset, and generated the training and testing datasets. She refactored and modified the existing codebase to enable end-to-end training and evaluation using nuScenes data. Additionally, she produced inference results on the test set and contributed to the part analysis of the model's performance.

Kamal Elmallah contributed to model architecture design, along with loss function strategy. He also established and contributed evaluative methods. Additionally, he contributed to the input data formation and strategy.

In addition to collaborative mode analysis, Bryan Pineda contributed by sourcing and setting up access to nuScenes dataset for the team, enabling the use of real-world multimodal data. He also conducted an initial literature survey to identify potential modeling strategies and guided architectural choices based on prior work. Furthermore, he explored and began prototyping a custom preprocessing pipeline tailored to panoramic camera and LiDAR inputs, although this component was ultimately not deployed in the final model.

References

[1] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[2] 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*, pages 11621–11631, 2020.

[3] 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*, 2024.

[4] F. Codevilla, E. Santana, A. M. López, and A. Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9329–9338, 2019.

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

[6] B. B. Elallid, N. Benamar, A. S. Hafid, T. Rachidi, and N. Mrani. A comprehensive survey on the application of deep and reinforcement learning approaches in autonomous driving. *Journal of King Saud University-Computer and Information Sciences*, 34(9):7366–7390, 2022.

[7] Y. Jeong, S. Kim, and K. Yi. Surround vehicle motion prediction using lstm-rnn for motion planning of autonomous vehicles at multi-lane turn intersections. *IEEE Open Journal of Intelligent Transportation Systems*, 1:2–14, 2020.

[8] J. Mao, S. Shi, X. Wang, and H. Li. 3d object detection for autonomous driving: A comprehensive survey. *International Journal of Computer Vision*, 131(8):1909–1963, 2023.

[9] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

[10] A. Prakash, A. Behl, E. Ohn-Bar, K. Chitta, and A. Geiger. Exploring data aggregation in policy learning for vision-based urban autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11763–11773, 2020.

[11] A. Prakash, K. Chitta, and A. Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7077–7087, 2021.

[12] S. Teng, X. Hu, P. Deng, B. Li, Y. Li, Y. Ai, D. Yang, L. Li, Z. Xuanyuan, F. Zhu, et al. Motion planning for autonomous driving: The state of the art and future perspectives. *IEEE Transactions on Intelligent Vehicles*, 8(6):3692–3711, 2023.

[13] D.-F. Xie, Z.-Z. Fang, B. Jia, and Z. He. A data-driven lane-changing model based on deep learning. *Transportation Research Part C: Emerging Technologies*, 106:41–60, 2019.

[14] Z. Xu, Y. Liu, Y. Sun, M. Liu, and L. Wang. Rngdet++: Road network graph detection by transformer with instance segmentation and multi-scale features enhancement. *IEEE Robotics and Automation Letters*, 8(5):2991–2998, 2023.

[15] 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*, 2024.