Kyber-E2E submission to CVPR's CARLA Autonomous Driving Challenge

Mohammed Elmahgiubi Fazel Arasteh Wang Zhitao Yang Zhou Yuanxin Zhong
Hamidreza Mirkhani Weize Zhang Zhan Qu Zizhao Huang
Kasra Rezaee
Noah's Ark Lab, Huawei Technologies
mohammed.elmahgiubi1@huawei.com

Abstract

In this paper we present the architecture of the Kyber-E2E submission to CARLA Autonomous Driving (AD) challenge at CVPR 2024. Our solution consists of perception and planning modules that are independently trained. In addition, we employed handcrafted modules for localization and control to form the full AD pipeline. We utilized a meticulously hand-crafted privileged agent to collect data for training our perception and planning modules. Although not perfect, the privileged agent successfully handled all scenarios in CARLA Leaderbaord 2.0, providing a reliable dataset for training. The perception module was trained in a supervised fashion from privileged information. The planning module was initially trained using imitation learning to replicate the handcrafted privileged agent, and later fine tuned in a replay-based simulation environment.

1. Introduction

The CARLA Autonomous Driving (AD) challenge aims to advance autonomous vehicle research and development by testing their performance in complex traffic scenarios. The 2024 CARLA AD challenge adopts Leaderboard 2.0 as the evaluation framework, introducing a significant evolution from Leaderboard 1.0. Leaderboard 2.0 introduces increased complexity by incorporating demanding scenarios, such as navigating around vehicles with open doors, yielding to emergency vehicles (e.g., firetrucks, police, and ambulances) approaching from behind, and managing highly adversarial situations like pedestrians crossing from behind an obstruction or vehicles violating traffic rules. The agent is expected to handle these challenges effectively.

This report provides an overview of the Kyber-E2E submission to the CARLA AD challenge. Our solution includes two trained modules (perception and planning) and two handcrafted modules (localization and control). We also leverage pre-trained object detection and visual question answer modules for detecting traffic signs and signals to complement our perception.

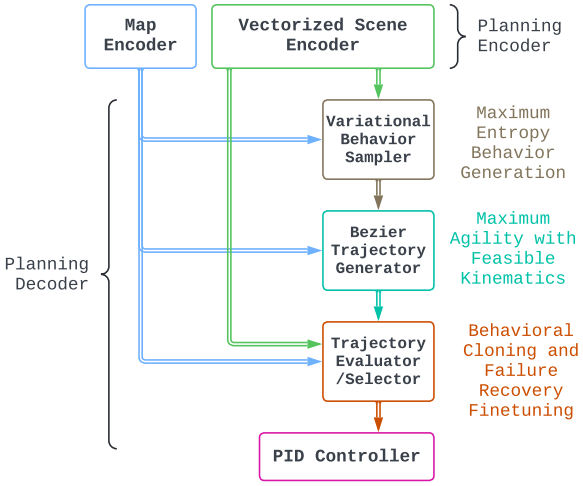

Figure 1. Overview of our Kyber end-to-end architecture, for CARLA Leaderboard 2.0 competition.

In section 2 we present details about the design and development of each component. Section 3 discusses the data used for training the models. In section 4, We discuss the training process and some empirical results. Finally, the conclusion and limitations of our work are provided in section 5.

2. Methodology

Figure 1 provides an overview of our Kyber-E2E architecture, which includes the following modules: sensing, perception, planning, and control. The subsequent sections will discuss these modules in more detail.

2.1. Sensing

Our sensing module consists of the following sensors:

- Six RGB cameras forming a panoramic camera set.

- One front-facing RGB camera, dedicated to traffic sign and signal detection.
- A 360-degree LiDAR for object detection.
- GNSS, IMU, and odometer to estimate the ego vehicle's state.
- An OpenDrive map to extract reference path and enhance perception output.

Compared to our solution in the previous challenge [11], we have added panoramic camera input. This enhancement increases the object detection range and allows for the processing of nuance information in the images, such as vehicle turn signals and siren lights.

2.2. Perception

The sensor information and map data are passed to the perception module to perceive the driving environment. The primary components of the perception module are the object detection and traffic signs understanding models. Our updated object detection module not only classifies objects and determines their bounding boxes but also predicts their velocity and future trajectories. The traffic sign understanding module, based on transformer models, [11], detects traffic signs and signals. For simplicity, we use map information in both the perception and planning stages. Using these two modules, the agent can drive smoothly in CARLA, though some post-processing of the perception results is applied to enhance the agent's performance.

2.2.1 Signs Detection and Recognition

We utilize the same approach, known as Language Aided Perceptions (LAPS), as in our previous model [11] to detect traffic signs and lights using off-the-shelf Transformer-based models. Specifically, we first employ the DETR [2] model to extract traffic signs from a high-resolution front-facing image. Then, we use the ViLT [5] model to query cropped images, extracting information such as traffic light states and speed limit values.

2.2.2 Object Detection and Prediction

Dynamic and static objects are detected using a VAD[4]-like network, which outputs object information in a vectorized format. The model consists of the encoders for sensor data, PV-to-BEV transformation module, and a deformable DETR head that outputs the structured detection results. The key components of the detection module are:

Encoder for image and point cloud: Given multi-frame and multi-view image inputs, ResNet50-FPN[7] serves as the default backbone network for encoding image features. The input image size is 640x360, which is sufficient for our perception range in CARLA while maintaining acceptable computational and memory budgets. The point cloud data is initially voxelized and processed by a point pillar network. After the PV-to-BEV transformation, the point cloud features are fused with the image features.

View transformation and multimodal-temporal fusion: We utilize a set of BEV queries to project the multi-view image features to BEV features, as done in BEVFormer[6]. To fuse multimodal information, the generated BEV features are concatenated with the point cloud features along the channel dimension and processed through an MLP layer. Subsequently, temporal self-attention is applied to aggregate the multimodal BEV features from the previous frame in a recurrent manner, accounting for ego motion to ensure spatial alignment. The range of BEV features is set to $x \in [-30, 70], y \in [-30, 30]$ meters, which has proven effective in most CARLA scenarios.

Detection head: As in view transformation, a set of obstacle queries is initialized and attends to the shared BEV feature map to learn features of agents and static obstacles, such as normal cars, emergency vehicles, open-door cars, pedestrians, bicycles, traffic warnings, and traffic cones. To increase efficiency, deformable attention [10] is employed. Finally, an MLP-based decoder head extracts object attributes, including classification scores, locations, orientations, and velocities, from the agent queries. Our perception network can provide all necessary object information for the downstream planner in an end-to-end fashion, without the need for post-processing.

2.2.3 Post-processing

In our submissions, we utilize several post-processing methods to enhance the perception outputs, including the following ones:

Object tracking: To refine orientations and speed estimations of the objects, the information of detected dynamic objects are passed through a UKF-based object-tracking module with imbalanced linear-sum assignment. The predicted trajectories are then fused with constant-velocity predictions for the planner module, in the downstream.

Map compliance: To identify the stop-lines and provide other necessary map-related information to the planner, We match traffic signs to map elements.

2.3. Planning and Control

We employ a planning module, which is trained to imitate handcrafted privileged agent's trajectories. The generated trajectories are passed though a calibrated controller that to output actions in the form of throttle/brake and steering commands.

2.3.1 Planner

The planner's backbone is a variation of the VectorNet [3]. The input to the planner's encoder is a number of candidate trajectories, which are generated using predefined parameters and a Bezier curve generator. The output of the encoder is used to learn a scoring module, identifying the best trajectory among the generated candidates. Additionally, a number of auxiliary tasks are employed in training the planner, enhancing the model's causal confusion. In general, our planner is trained using the following two complementary paradigms:

Imitation learning: The planner is initially trained to imitate the expert trajectories. At this stage, we leverage supervised learning to train motion planning and scoring modules, selecting the the trajectory that is closest to the expert's trajectory.

Closed-loop fine tuning: To address the known issue of distributional shift in imitation learning [8, 9], we adapt a closed-loop training paradigm, learning from mistakes (LfM) [1], to enhance the performance of the score module. We deploy the planner in a replay-based simulator, where other objects follow their positions from the recorded data, and use some basic rules to detect failures such as collision and driving off the road. When a failure happens, the data is collected and added to a new dataset, by which the score module is further trained to assign lower scores to those trajectories. Unlike other similar approaches such as DAgger [9], our approach does not require an expert to annotate the correct actions in such cases and can merely learn from its failures.

2.3.2 Controller

The controller module is decoupled into lateral controller and longitudinal controller, which both are classic PID controllers. The lateral controller takes heading error as input and outputs steering angle. The longitudinal controller takes acceleration error as input and outputs an acceleration. Using a calibrated feed-forward lookup table, throttle/brake command is generated from the output acceleration.

3. Data Collection

3.1. Handcrafted Privileged Agent

To the best of our knowledge, no reliable autopilot agent exists that can handle intricate Leaderbaord 2.0 scenarios. Therefore, We utilize a handcrafted privileged agent [11] that can handle all scenarios from Leaderboard 2.0.

3.2. Data

Our solution employs learning-based algorithms as the foundation for both the perception and planning modules. The training of these modules is achieved through data collection from CARLA Leaderboard 2.0. The data used for training the perception module is collected using Leaderbaord routes in addition to open-source data from Leaderbaord 1.0 (randomly generated routes from Town1 to Town10). It consists of 7 hours of driving frames with per-frame sensor data, ego state, and annotations. During the data collection, the surrounding traffic, pedestrians and weather are randomized. The ratio of the data with different sources and scenarios is carefully maintained to help the perception module work better on rare objects including traffic signs and cones.

The data for planning is also a combination of open-source data from Leaderbaord 1.0 and data collected from Leaderbaord 2.0 using our privileged agent. In total, we used about 10 hours of driving data to train the planner.

3.3. Scenario generation

We developed an automated scenario generation module designed to segment each route in Leaderbaord 2.0 into several short-duration segments each consisting of one scenario. Compared to collecting data on full routes, using data from short segments allow us to collect a balanced dataset (and modify the balance for perception and planning as needed) and efficiently remove failing scenarios from dataset.

4. Experiments and result

4.1. Privileged Agent

Our handcrafted privileged agent achieves an overall score of 27.25 on the Leaderbaord 2.0 validation routes. However, this model is quite sensitive to data quality, and noise in input data can significantly degrade its performance. In the absence of privileged data, and using a noisy perception, this agent achieves an overall score of 2-3 on the validation routes. It is worth noting that this result is based on our previous perception module and with the revised perception module the score is expected to be higher, but we have not evaluated that. Please refer to [11] for more details.

4.2. Perception Performance

The object detection model is trained on the perception dataset for roughly 50 epochs and it yields a decent performance on a validation set collected from Town 13. Please find the detailed performance reported in Table 1.

4.3. Planner Performance

We trained the planning model for 30 epochs with imitation learning and another 100 epochs with failures encountered in closed-loop simulation. At every closed-loop training epoch, we sample 100 min-scenarios (about 15 seconds) and evaluated the planner and collect mistakes in a second

Table 1. Evaluation of object detection and prediction performance. Note that the detection for the open-door and emergency vehicles are not included in this table, as they are not well represented in the collected evaluation set.

Class	Detection			Motion		
	AP	ATE	AVE	ADE	FDE	MR%
Car	0.57	0.113	0.582	0.69	1.09	11.5
Pedestrian	0.31	0.096	0.003	0.11	0.11	0.00
Bicyclist	0.72	0.071	0.283	-	-	-
Traffic Cone	0.38	0.180	0.017	-	-	-
Traffic Sign	0.81	0.088	0.039	-	-	-

Table 2. Evaluation of the submitted agent on CARLA official Leaderbaord 2.0 Validation routes

Team	Driving Score	Route Completion	Infraction Penalty
Kyber-E2E	6.816	13.686	0.510

Table 3. Evaluation of the submitted agent on CARLA official Leaderbaord 2.0 hidden Test routes

Team	Driving Score	Route Completion	Infraction Penalty
Tf++	5.564	11.815	0.467
Kyber-E2E (ours)	5.467	15.895	0.337
CarLLaVA	5.040	15.761	0.431
greatone	2.168	10.781	0.368
CaRINA modular	1.140	3.653	0.460

dataset. Then planner is trained with both expert data (imitation learning) and mistake data (Learning from Mistakes). Unlike the handcrafted agent that is sensitive to the perception noise, we found our ML-based E2E planner to not be affected by perception noise as much and the performance hit without privileged information much lower. We tested our submission on the Leaderbaord 2.0 validation routes and its performance is summarized in Table 2. Additionally, our latest submission to the online leaderboard is shown in Table 3.

5. Conclusion and Limitations

We introduced Kyber-E2E solution for the 2024 CARLA AD challenge, map track. Our result show that an agent constructed from perception and planning modules trained independently that use a vectorized representation in between can yield reasonable performance. One limitation of our submission is limited performance of the perception module for speed estimation that required use of a tracking module and kalman filter to effectively estimate the speeds. Additionally, we plan to incorporate traffic sign detection in our future perception module.

References

[1] Fazel Arasteh, Mohammed Elmahgiubi, Behzad Khamidehi, Hamidreza Mirkhani, Weize Zhang, and Kasra Rezaee. Learning from mistakes: a weakly-supervised method for mitigating the distribution shift in autonomous vehicle planning, 2024. 3

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 2

[3] 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*, pages 11525–11533, 2020. 3

[4] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8350, 2023. 2

[5] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision, 2021. 2

[6] 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 *European conference on computer vision*, pages 1–18. Springer, 2022. 2

[7] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 2

[8] Stephane Ross and Drew Bagnell. Efficient Reductions for Imitation Learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, pages 661–668, 2010. 3

[9] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A Reduction of Imitation Learning and Structured Prediction to No-regret Online Learning. In *Proceedings of the international conference on artificial intelligence and statistics (AISTATS 2011)*, pages 627–635, 2011. 3

[10] Zhuofan Xia, Xuran Pan, Shiji Song, Li Erran Li, and Gao Huang. Vision transformer with deformable attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4794–4803, 2022. 2

[11] Weize Zhang, Mohammed Elmahgiubi, Kasra Rezaee, Behzad Khamidehi, Hamidreza Mirkhani, Fazel Arasteh, Chunlin Li, Muhammad Ahsan Kaleem, Eduardo R Corral-Soto, Dhruv Sharma, et al. Analysis of a modular autonomous driving architecture: The top submission to carla leaderboard 2.0 challenge. *arXiv preprint arXiv:2405.01394*, 2024. 2, 3