CarLLaVA: Vision language models for camera-only closed-loop driving

Katrin Renz[1,2,3*] Long Chen[1] Ana-Maria Marcu[1] Jan Hünermann[1]
Benoit Hanotte[1] Alice Karnsund[1] Jamie Shotton[1] Elahe Arani[1] Oleg Sinavski[1]

[1] Wayve [2] University of Tübingen [3] Tübingen AI Center

Abstract

In this report, we present CarLLaVA, a Vision Language Model (VLM) for autonomous driving, developed for the CARLA Autonomous Driving Challenge 2.0. CarLLaVA uses the vision encoder of the LLaVA VLM and the LLaMA architecture as backbone, achieving state-of-the-art closed-loop driving performance with only camera input and without the need for complex or expensive labels. Additionally, we show preliminary results on predicting language commentary alongside the driving output. CarLLaVA uses a semi-disentangled output representation of both path predictions and waypoints, getting the advantages of the path for better lateral control and the waypoints for better longitudinal control. We propose an efficient training recipe to train on large driving datasets without wasting compute on easy, trivial data. CarLLaVA ranks 1st place in the sensor track of the CARLA Autonomous Driving Challenge 2.0 outperforming the previous state-of-the-art by 458% and the best concurrent submission by 32.6%.

1. Introduction

The trend in autonomous driving is shifting towards end-to-end solutions, showed by recent advances in industry [32] and the state-of-the-art performance on the CARLA Leaderboard 1.0 [6, 15, 26, 29, 38]. Most of the top-performing entries on the CARLA Leaderboard 1.0 [1] rely on expensive LiDAR sensors, with the exception of TCP [38], which employs a camera-only approach. Additionally, multi-task learning has emerged as a common strategy for enhancing performance [9]. However, this requires access to labels, such as BEV semantics, depth, or semantic segmentation, which are expensive to obtain in the real world. This makes it hard to transfer insights from research using simulators to real world driving in a scalable and cost-efficient way. CarLLaVA in contrast only relies on commonly available and easy to obtain driving data such as camera images and driving trajectory and is a camera only method.

Additionally, most state-of-the-art CARLA methods use ResNet-style backbones pretrained on ImageNet [15, 26, 29, 38]. However, recent progress in pretraining techniques, such as CLIP [23], MAE [13], and DINO, have demonstrated the advantages of using Vision Transformers (ViTs) [30] over traditional CNN-encoders for improved feature learning. Moreover, state-of-the-art VLMs [8, 17, 20] that fine-tune the CLIP encoder exhibit nuanced image understanding, indicating the existence of strong vision features. CarLLaVA makes use of this by using the vision encoder of LLaVA-NeXT [19–21] which is pre-trained on internet-scale vision-language data. While the size of modern VLMs could be viewed as a concern for inference time when deployed on real vehicles, several recent works showed that this is a solvable engineering problem [2, 3, 34].

In this technical report, we describe the details of our driving model CarLLaVA, which includes the following properties and advantages: **Camera only without expensive labels**: Our method only uses camera input, eliminating the need for additional expensive labels such as Bird's Eye View (BEV), depth, or semantic segmentation. This label-free approach reduces dependency on extensive labeled datasets, making deployment on real cars more feasible. **Vision-Language Pretraining**: Our approach leverages a vision encoder pre-trained on internet-scale vision-language data. We demonstrate that this pretraining can be effectively transferred to the task of driving, resulting in improved driving performance compared to training from scratch on driving data. **High-resolution input**: We noticed that the default resolution of the CLIP vision encoder is not sufficient for quality driving. Similar to LLaVA[21], we split input images into patches to allow the VLM access smaller details in the driving images such as distant traffic lights and pedestrians. In contrast to LLaVA we do not use the small resolution global patch to reduce the number of tokens. **Efficient Training Recipe**: We propose an efficient training recipe that makes more use of interesting training samples, significantly reducing training time. **Semi-Disentangled Output Representation**: We propose a semi-disentangled representation with both time-conditioned waypoints and space-conditioned path waypoints, leading to better control.

*Work done while interning at Wayve.

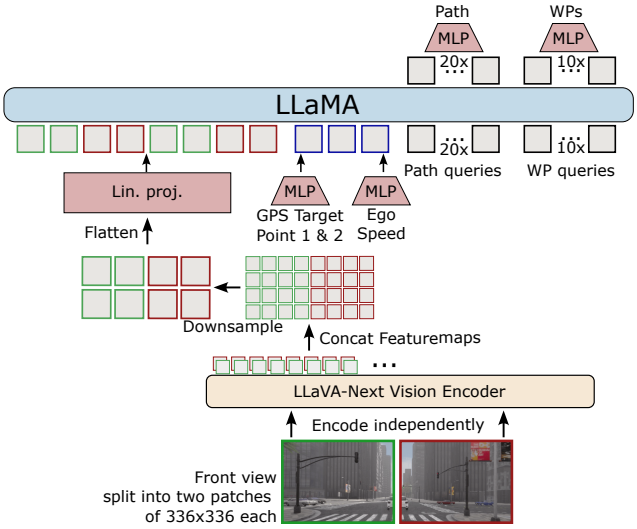

Figure 1. **CarLLaVA base model architecture. (C1T1)** The images are split in two, and each split is independently encoded and then concatenated, downsampled and projected into a pre-trained large language model. The output utilises a semi-disentangled representation with both time-conditioned waypoints and space-conditioned path waypoints for improved lateral control.

2. Method

In the following sections, we provide a comprehensive overview of our architecture and training methodology.

Task. The objective is to reach a specified target location on a 10x10 km^2 map while passing predetermined intermediate target points. The map includes diverse environments such as highways, urban streets, residential areas, and rural settings, all of which must be navigated under various weather conditions, including clear daylight, sunset, rain, fog, and nighttime scenarios. Along the way the agent must manage various complex scenarios such as encountering pedestrians, navigating parking exits, executing unprotected turns, merging into ongoing traffic, passing construction sites or avoiding vehicles with opening doors.

Architecture. An overview of our base architecture can be seen in Fig. 1.

Input/Output Representation. The model inputs include camera images, the next two target points, and the ego vehicle's speed. We tested several configurations: (1) the base model (C1T1) with a single front view image, (2) the temporal model (C1T2) which includes image features from the previous timestep, and (3) the multi-view model (C2T1) which adds a low-resolution rear-view camera to the high-resolution front view. For the output, we use a semi-disentangled representation with both time-conditioned waypoints with a PID controller for longitudinal control and space-conditioned path waypoints with a PID controller for lateral control. Early experiments with entangled waypoints led to steering errors, especially during turns or when swerving around obstacles. By using path waypoints, we achieve denser supervision, as we also predict the path when the vehicle is stationary, leading to improved steering behaviour. For longitudinal control we use standard time-conditioned waypoints to make use of the better collision avoidance compared to directly predicting control [37]. We also experimented with target speed classification and GRUs, but these methods did not perform as well, although we lack official performance metrics.

HD-Vision Encoder. To encode the camera images, we use the LLaVA-NeXT vision encoder, specifically the CLIPViT-L-336px model, which is the highest resolution trained CLIP model. High-resolution images are crucial for driving because important information, such as traffic lights at large intersections, may only be visible in a few pixels. To leverage CLIP pre-training at higher resolutions than 336x336, we use LLaVA's *anyres* technique [21]. We divide high-resolution images into multiple large patches of up to 336x336 pixels, encoding each independently, and then concatenating the resulting features in spatial dimension to form a single large feature map for the original image. Using a VLM not only provides strong features, but also offers the advantage of easily query the VLM to identify what information are captured in the image features. More specifically, we queried the VLM for example for the state of traffic lights at different input resolutions to determine the optimal resolution and therefore the number of patches.

Adapter. To reduce computation overhead due to the nature of the quadratic complexity of the LLaMA transformer, we downsample the feature map to half the number of tokens. After flattening, we employ a linear projection layer to map the vision features to the embedding space of the language model. To encode the target points and ego speed, we utilize a multi-layer perceptron (MLP) following a normalization layer. Additionally, we add camera encodings for the different views (model C2T1) and temporal encodings when using images from multiple time steps (only for model C1T2).

LM-Decoder. We use the LLaMA architecture as a decoder. In addition to the sensor input tokens, we use learnable queries to generate the path and waypoints. An MLP on top of the output features generates waypoint differences. The cumulative sum of these differences yields the final waypoints, which are supervised during training using mean squared error (MSE) loss. For our preliminary results on generating language explanations we auto-regressively sample the language explanation after generating the path and waypoints. During training we feed the tokenized explanations and use a standard language modelling (LM) loss. We use the tokenizer and LM-head of the pretrained Tiny-LLaMA model.

Efficient training of large models. Our models have

between 350M and 1.3B parameter. To finetune these large models on our task we rely on models pretrained on internet-scale data, a large dataset and an efficient training recipe which is described in the following.

Dataset We utilize the privileged rule-based expert *PDM-light* [4] to collect a dataset. We divide the official CARLA routes of Town 12 and Town 13 into shorter segments centered around scenarios to reduce trivial data (e.g., driving straight without any hazardous events) and simplify data management. We use short routes with a single scenario as proposed by[9, 18], however with the introduction of Leaderboard 2.0, the maximum distance between target points increased from 50 meters to 200 meters. The short routes often fall within this distance, causing a distribution shift, as the next target point is the end of the route (i.e, closer than 200m) rather than the position that would be used when having long routes. Consequently, we employ a second set of routes featuring three scenarios per route. To ensure balance, we adjust the number of routes per scenario and apply random weather augmentation and modify the parameter *distance* for scenarios by ±10%. Overall, we collect 2.9 million samples at 5 fps.

For the language generation experiment we use the logic of the rule-based expert to generate explanations. More precisely, we use the leading object obtained from the experts' Intelligent Driver Model (IDM)[33] as well as information about changing the path to swerve around objects. In addition, we use heuristics based on the ego waypoints to distinguish between driving intentions like starting from stop or keep driving at the same speed. As this experiment is only intended to showcase the potential of using LLMs for driving, we do not add detailed statistics of the obtained labels and keep it for future work.

Buckets. The majority of driving involves straight, uneventful segments. To maximize the collection of interesting scenarios during data collection, we focus on capturing a diverse range of challenging situations. However, some ratio of easy and uneventful data is inevitable. Training models on the entire dataset revealed that straight driving without hazards is effectively learned in the early epochs, resulting in wasted compute in later epochs as the models continue to train on these uninteresting samples. To address this issue, we create data buckets containing only the interesting samples and sample from these buckets during training instead of the entire dataset. We use: (1) five buckets for different amount of acceleration and deceleration with one specifically for starting from stop, excluding samples with acceleration between -1 and 1, (2) two buckets for steering, excluding samples for going straight, (3) three buckets for vehicle hazard with vehicles coming from different directions, (4) one for stop sign, red light and walker hazards each, (5) one bucket for swerving around obstacles and (6) one bucket that samples from the whole dataset to keep a small portion of uneventful data such as driving straight. This approach reduces the number of samples per epoch to 650,000.

	Method	Sensors	Aux. Labels	DS ↑	RC ↑	IS ↑
Map	CaRINA modular	L+C+M	OD	1.14	3.65	0.46
	greatone	undisclosed	undisclosed	2.17	10.78	0.37
	Kyber-E2E	L+C+R+M	IS, OD	3.47	8.48	**0.50**
	TF++	L+C	SS, D, OD, BS	**5.56**	**11.82**	0.47
Sensor	CARLA	priv.	priv.	0.25	15.20	0.10
	Zero-shot TF++	L+C	SS, D, OD, BS	0.58	8.53	0.38
	CaRINA hybrid	L+C	IS, OD	1.23	9.56	0.31
	TF++	L+C	SS, D, OD, BS	5.18	11.34	**0.48**
	CarLLaVA (ours)	C	-	**6.87**	**18.08**	0.42

Table 1. **Leaderboard 2.0 Results.** CarLLaVA achives state-of-the-art performance on the leaderboard. Legend: L: Lidar, C: Camera, R: Radar, M: Map, priv: privileged, OD: Object Detection (3D position and pose), IS: Instant Segmentation, SS: Semantic Segmentation, D: Depth, BS: BEV semantics.

3. Experiments

Benchmarks. *Leaderboard2.0.* We use the official test server with secret routes under different weather conditions. *10xShort.* For the models where we could not get Leaderboard results, we use a local evaluation on short routes with one scenario per route to evaluate the models ability to solve each scenario type. We use maximum 10 routes per scenario which are randomly sampled from the whole set.

Metrics. We report the official CARLA metrics, Driving Score (DS), Route Completion (RC) and Infraction Score (IS). DS is calculated in a way that the reduction due to infractions does not linearly correlate with the increase in DS due to higher RC (i.e., with a constant infraction per km the DS gets much worse for higher RC for models that can solve the scenarios below a certain percentage). Forcing the agent to stop a route early can maximize DS.

Implementation Details. We refer to the supplementary for implementation details.

Results. *Leaderboard state of the art.* We present the official Leaderboard results in Tab. 1. With our base-model CarLLaVA C1T1 we outperfrom the state of the art (5.18 vs 6.87 DS). However, we observed a high variance on the Leaderboard score, detailed results on mean and standard deviation can be found in the supplementary (the official Leaderboard numbers are our first submissions of the models, the repetitions to calculate mean and std happened after the challenge deadline). It is also noteworthy that, to the best of our knowledge, our model is the only model on the leaderboard working only with camera images and without the usage of additional auxiliary labels (note: for the new entry *greatone* we do not know what their method is).

Output representation. Tab. 2a compares the DS on the Leaderboard for the different output representations. As the goal of the additional path prediction is improved lateral

	DS ↑	Stat ↓
WPs	3.21	0.68
+Path	4.49	0.0

(a) **Output.**

	DS ↑
LLaVA	6.87
- pretraining	0.45
Resnet-34	2.71

(b) **Vision encoder**.

	DS ↑
1300	3.93
1800	4.49
2100	6.87
2400	6.35

(c) **Early stopping**.

Table 2. Ablations of different parts of our model, showcasing the superiority of the semi-disentangled output representation and the large impact of the correct threshold for early stopping. The score of the default configuration is highlighted in gray. All numbers are official Leaderboard scores.

control, we also report the collisions with static layout as this is mainly caused due to bad steering. With the semi-disentangled representation we can reduce the layout collision from 0.68 to 0.0 showcasing the strength of additional path predictions.

Vision-Language and CLIP pretraining. We ablate the pretraining of the vision encoder and train the same model from scratch. Tab. 2b '-pretraining' shows that the pretraining stage is essential for good driving performance (more tuning of the training hyperparameters can further improve the performance but is unlikely to reach the performance of the pretrained model). Additionally, we show a comparison to the widely used Resnet-34 pretrained on ImageNet. The decreased performance (2.71 vs. 6.87 DS) indicates the importance of the larger ViT and the internet-scale image-language pretraining.

Early stopping. We ablate the thresholds for the early stopping as it is not trivial to calculate the perfect trade-off as the routes and density of scenarios are secret (however a rough function of the expected DS can be caluculated which we used to get a rough idea). Tab. 2c shows the Leaderboard DS for a given travelled distance in meters. This hyperparameter has a big impact on the final score.

Preliminary Results. In addition to our ablations we show preliminary results to showcase the potential to extend to multiple views and temporal input as well as scaling our base model. Detailed results can be found in the supplementary. *Language explanations.* With the additional language training our model is able to produce commentary that comments the current driving behaviour (Fig. 2). This is not intended as an actual explanation as the training misses an important grounding step (i.e., commentary is not always aligned with the actions the model takes). We leave this for future work.

Failure cases. The most common failure cases of our model are rear end collision, which can be reduced by using the temporal input of the C1T2 model and maneuver like merging especially in high speeds.

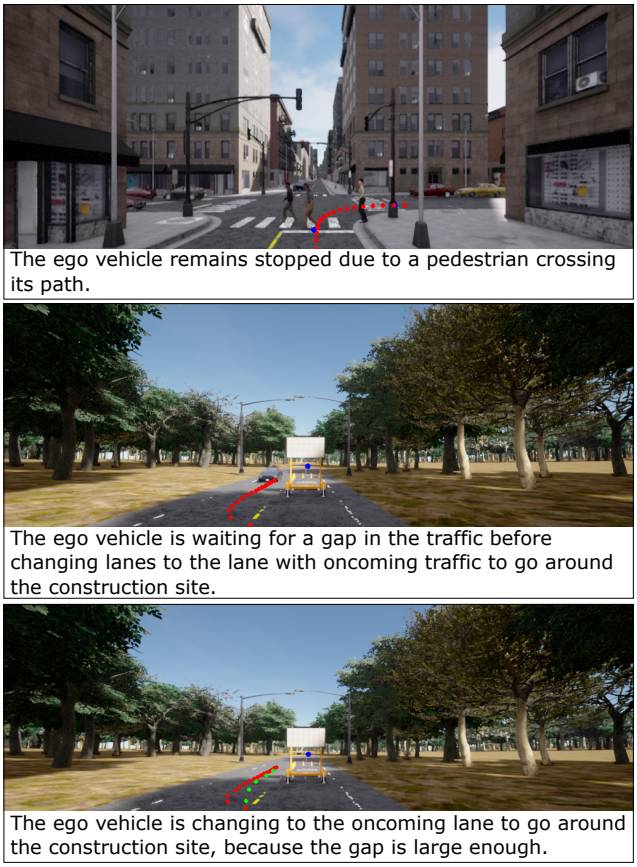

Figure 2. **Qualitative examples of generated language.** Red: predicted path, Green: predicted waypoints, Blue: Target Points

4. Conclusion

In this report, we present CarLLaVA the winning entry in the CARLA Autonomous Driving Challenge 2.0 2024, which leverages vision-language pretraining and uses only camera images as input. By utilizing a semi-disentangled output representation and an efficient training approach, CarLLaVA demonstrates superior performance in both lateral and longitudinal control. Its ability to operate without expensive labels or sensors makes it a scalable and cost-effective solution. The results indicate a substantial improvement over previous methods, showcasing the potential of vision-language models in real-world autonomous driving applications.

Acknowledgements. We thank Kashyap Chitta, Julian Zimmerlin, Jens Beißwenger, Bernhard Jäger and Andreas Geiger for valuable discussions and help with the expert. We also thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting K. Renz.

References

[1] Carla autonomous driving leaderboard. https://leaderboard.carla.org/, 2020. 1

[2] Lingo-2: Driving with natural language. https://wayve.ai/thinking/lingo-2-driving-with-language/, 2024. 1

[3] Lambda: The nuro driver's real time. https://medium.com/nuro/lambda-the-nuro-drivers-real-time-language-reasoning-model-7c3567b2d7b4, 2024. 1

[4] Jens Beißwenger. Pdm-lite: A rule-based planner for carla leaderboard 2.0. Technical report, University of Tübingen, 2024. 3

[5] Dian Chen and Philipp Krähenbühl. Learning from all vehicles. In *CVPR*, 2022. 7

[6] Dian Chen and Philipp Krähenbühl. Learning from all vehicles. In *CVPR*, 2022. 1

[7] Long Chen, Oleg Sinavski, Jan Hünermann, Alice Karnsund, Andrew James Willmott, Danny Birch, Daniel Maund, and Jamie Shotton. Driving with llms: Fusing object-level vector modality for explainable autonomous driving, 2023. 7

[8] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks, 2024. 1

[9] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, , and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE T-PAMI*, 2023. 1, 3

[10] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *PAMI*, 2023. 7

[11] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024. 7

[12] Daocheng Fu, Xin Li, Licheng Wen, Min Dou, Pinlong Cai, Botian Shi, and Yu Qiao. Drive like a human: Rethinking autonomous driving with large language models, 2023. 7

[13] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. *arXiv.org*, 2111.06377, 2021. 1

[14] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *CoRL*, 2021. 7

[15] Bernhard Jaeger, Kashyap Chitta, and Andreas Geiger. Hidden biases of end-to-end driving models, 2023. 1, 7

[16] Xiaosong Jia, Penghao Wu, Li Chen, Jiangwei Xie, Conghui He, Junchi Yan, and Hongyang Li. Think twice before driving: Towards scalable decoders for end-to-end autonomous driving. In *CVPR*, 2023. 7

[17] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023. 1

[18] Qifeng Li, Xiaosong Jia, Shaobo Wang, and Junchi Yan. Think2drive: Efficient reinforcement learning by thinking in latent world model for quasi-realistic autonomous driving (in carla-v2), 2024. 3

[19] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023. 1

[20] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 1

[21] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, 2024. 1, 2

[22] Jiageng Mao, Yuxi Qian, Hang Zhao, and Yue Wang. GPT-Driver: Learning to drive with gpt. *arXiv preprint arXiv:2310.01415*, 2023. 7

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763, 2021. 1

[24] Hao Sha, Yao Mu, Yuxuan Jiang, Li Chen, Chenfeng Xu, Ping Luo, Shengbo Eben Li, Masayoshi Tomizuka, Wei Zhan, and Mingyu Ding. LanguageMPC: Large language models as decision makers for autonomous driving. *arXiv preprint arXiv:2310.03026*, 2023. 7

[25] Hao Shao, Letian Wang, RuoBing Chen, Hongsheng Li, and Yu Liu. Safety-enhanced autonomous driving using interpretable sensor fusion transformer. In *CoRL*, 2022. 7

[26] Hao Shao, Letian Wang, RuoBing Chen, Hongsheng Li, and Yu Liu. Safety-enhanced autonomous driving using interpretable sensor fusion transformer, 2022. 1

[27] Hao Shao, Yuxuan Hu, Letian Wang, Steven L. Waslander, Yu Liu, and Hongsheng Li. Lmdrive: Closed-loop end-to-end driving with large language models, 2023. 7

[28] Hao Shao, Letian Wang, Ruobing Chen, Steven L. Waslander, Hongsheng Li, and Yu Liu. Reasonnet: End-to-end driving with temporal and global reasoning. In *CVPR*, 2023. 7

[29] Hao Shao, Letian Wang, Ruobing Chen, Steven L. Waslander, Hongsheng Li, and Yu Liu. Reasonnet: End-to-end driving with temporal and global reasoning, 2023. 1

[30] Gilad Sharir, Asaf Noy, and Lihi Zelnik-Manor. An image is worth 16x16 words, what is a video worth? In *ICLR*, 2021. 1

[31] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering, 2023. 7

[32] Brad Templeton. Tesla, waymo, nuro, zoox and many others embrace new ai to drive. *Forbes*, 2024. Accessed: 02 June 2024. 1

[33] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805–1824, 2000. 3

[34] Tsun-Hsuan Wang, Alaa Maalouf, Wei Xiao, Yutong Ban, Alexander Amini, Guy Rosman, Sertac Karaman, and Daniela Rus. Drive anywhere: Generalizable end-to-end

autonomous driving with multi-modal foundation models. *arXiv preprint arXiv:2310.17642*, 2023. 1

[35] Wenhai Wang, Jiangwei Xie, ChuanYang Hu, Haoming Zou, Jianan Fan, Wenwen Tong, Yang Wen, Silei Wu, Hanming Deng, Zhiqi Li, Hao Tian, Lewei Lu, Xizhou Zhu, Xiaogang Wang, Yu Qiao, and Jifeng Dai. Drivemlm: Aligning multi-modal large language models with behavioral planning states for autonomous driving, 2023. 7

[36] Licheng Wen, Daocheng Fu, Xin Li, Xinyu Cai, Tao Ma, Pinlong Cai, Min Dou, Botian Shi, Liang He, and Yu Qiao. Dilu: A knowledge-driven approach to autonomous driving with large language models, 2023. 7

[37] Peng Wu, Xiaosong Jia, Li Chen, Junchi Yan, Hongyang Li, and Yu Qiao. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. In *NeurIPS*, 2022. 2, 7

[38] Penghao Wu, Xiaosong Jia, Li Chen, Junchi Yan, Hongyang Li, and Yu Qiao. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. In *NeurIPS*, 2022. 1

[39] Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kenneth KY Wong, Zhenguo Li, and Hengshuang Zhao. DriveGPT4: Interpretable end-to-end autonomous driving via large language model. *arXiv preprint arXiv:2310.01412*, 2023. 7

[40] Jimuyang Zhang, Zanming Huang, and Eshed Ohn-Bar. Coaching a teachable student. In *CVPR*, 2023. 7

APPENDIX

A. Related Work

Foundation models for driving. Recently, large language models (LLMs) have been integrated into driving systems to leverage their reasoning capabilities for addressing long-tail scenarios. Multi-modal LLM-based driving frameworks such as LLM-Driver [7], DriveGPT4 [39], and DriveLM [31] utilize foundation models with the inputs from different modalities for driving. GPT-Driver [22] and LanguageMPC [24] fine-tune ChatGPT as a motion planner using text. Knowledge-driven approaches [12, 36] are also adopted to make decisions based on common-sense knowledge and evolve continuously. However, most of these works have been evaluated primarily through qualitative analysis or in open-loop settings. The most similar works leveraging foundation models for closed-loop driving in CARLA are DriveMLM [35] and LMDrive [27], which utilize multi-modal LLMs. However, these approaches rely on image and LiDAR inputs with customized encoders, without leveraging the power of vision-language pretraining and focused on tasks like intruction following. In comparison we focus on pure closed-loop driving performance to provide a baseline that can solve basic driving behaviors to enable future research on VLMs for driving.

End-to-end closed-loop driving in CARLA. End-to-end training based on Imitation Learning (IL) is the dominant approach for state-of-the-art methods on the CARLA Leaderboard 1.0 [5, 15, 25, 37]. Those methods are mostly incorporate numerous auxiliary outputs and rely on expensive sensors like LiDAR. In contrast, we build a model that only relies on camera images and the driving trajectory.
The dominant output representation is predicting waypoints with a GRU and using PID-controllers for lateral and longitudinal control [5, 10, 15, 16, 25, 28, 37, 40]. TCP [37] showed that waypoints perform poorly in turns, but predicting direct control performs worse in avoiding collisions. They propose a situation-based fusion strategy of those representations. Interfuser [25] proposed predicting path waypoints together with a combination of forecasting and heuristics to obtain control. TF++ [15] uses path waypoints for lateral control and target speed classes for longitudinal control. In our work we leverage the path representation for improved steering together with the standard waypoints for longitudinal control avoiding heuristics or the need for predefined classes. Additionally directly predict the waypoints from the output features of the transformer without using GRU.

	$DS_S \uparrow$
50M	90.40
350M	92.49
1B pt LoRA	90.03
1B s LoRA	89.57

(a) **Scale**.

	$DS_S \uparrow$
default	90.40
+ temporal	90.37
+ back	88.81
- pretraining	75.43

(b) **Input**.

Table A.1. Further ablations of different parts of our model. The score of the default configuration is highlighted in gray. DS_S is performance on the *10xShort* benchmark.

B. Implementation Details

We use a learning rate of 3e-5 with a cosine annealing schedule. The batch size of our base model is 20, while for specific configurations, we use a batch size of 10 for C1T2 and a batch size of 12 for C2T1. The AdamW optimizer is employed with a weight decay of 0.1. Our vision encoder consists of 305 million parameters. We experiment with the LLaMA architecture in three configurations: LLaMA-50M, LLaMA-350M (both trained from scratch), and a 1B TinyLLaMA with LoRA finetuning [14], applied to all linear layers as demonstrated to be effective by QLoRA [11]. We apply the same data augmentation techniques as TF++ [15] but with more aggressive shift and rotation augmentation (shift: 1.5m, rot: 20 deg). Additionally, we add histogram enhancements to improve the contrast and quality of input images for night time driving. DeepSpeed v2 is utilized for optimizing training efficiency and memory usage. We train for 30 epochs. Our base model, C1T1, trains in approximately 27 hours using 8xA100 40GB GPUs. During inference we apply early stopping to counter the nature of DS described in the metric section. We track the travelled distance and stop driving after a specified distance when the steering angle is close to zero to prevent stopping in the middle of an intersection where it could happen that other vehicles crash into us.

C. Additional ablations

Leaderboard variance. We submitted our base model CarLLaVA C1T1 with an early stopping threshold of 2100 and 2400 three times to the leadboard to get an estimate of the evlauation variance. For the 2100 model we obtain the following scores: 5.5, 6.8 and 5.3 resulting in a mean DS of 5.87 with a standard deviation of 0.81. The base model with a threshold of 2400 obtained 6.3, 6.3 and 4.8 resuting in a mean of 5.8 with standard deviation of 0.87.
Scale. In an additional experiment we scale up the LLaMA architecture (Tab. A.1a). Training a 350M parameter model from scratch improves performance slightly. However scaling to 1B parameter and finetuning with LoRA resulted in worse performance for using a pretrained LLM (pt) and training from scratch (s). We suspect that this may be due

to the use of LoRA finetuning and not fully tuned hyperparameters, but further investigation is needed. This remains an interesting research question for future work.

Extending the input. To be able to fully solve autonomous driving, information from more than one camera (especially for camera-only arhcitectures) and temporal information are needed. In Tab. A.1b we show results for a model with temporal information and one with added back camera. Qualitative investigations showed improvements in the expected scenarios (less rear-end collisions for *+temporal* and improved lane-change behaviour for *+back*). Interestingly the overall score does not increase.