Supplementary Materials: Trajectory-guided Control Prediction for End-to-end Autonomous Driving: A Simple yet Strong Baseline

Penghao Wu[*†]
Shanghai AI Laboratory
Shanghai Jiao Tong University
wupenghaocraig@sjtu.edu.cn

Xiaosong Jia[*]
Shanghai AI Laboratory
Shanghai Jiao Tong University
jiaxiaosong@sjtu.edu.cn

Li Chen[*]
Shanghai AI Laboratory
lichen@pjlab.org.cn

Junchi Yan[†]
Shanghai Jiao Tong University
yanjunchi@sjtu.edu.cn

Hongyang Li
Shanghai AI Laboratory
Shanghai Jiao Tong University
lihongyang@pjlab.org.cn

Yu Qiao
Shanghai AI Laboratory
qiaoyu@pjlab.org.cn

In this Supplementary document, we first provide a detailed description of the dataset in Sec. A. Implementation and training details are in Sec. B. We show detailed infraction statistics for both leaderboard results and ablation studies, and qualitative results in Sec. C. Last, we discuss limitations, common failure cases, and possible future directions and potential social impact of our work in Sec. D.

A Dataset

A.1 Dataset Collection

We use CARLA 0.9.10.1 for data collection and testing. We use Roach [16] as the expert to collect data. In order to improve the obstacle avoidance ability of the expert, we additionally add a rule-based vehicle and pedestrian detector adopted from Transfuser [12] to avoid possible collisions. Each route is generated randomly with length ranging from 50 meters to 300 meters. We use the scenario configurations provided in [12]. We terminate each route if the expert makes a collision or runs a red light. Last few frames for such routes are discarded. The data samples are stored at 2HZ.

A.2 Dataset Statistics

Detailed statistics for each town and their descriptions are provided in Table 1. As stated in the main paper, we train on all eight towns for the leaderboard submission. For our ablation experiments, we train on four towns (Town01, Town03, Town04, and Town06) and test on the designed four routes with four different weathers in Town02 and Town05, as does in [3].

B Implementation Details

We use ResNet-34 [8] pretrained on ImageNet [6] as the image encoder. The size of the input image is 900×256 and the FOV of the camera is set as 100°. We choose K being 4 meaning four future steps at 2HZ are predicted for both the trajectory branch and the control branch. Detailed network structure is presented in Table 5. We follow the same PID setting as [5], where the PID parameters are exquisitely tuned, *i.e.*, K_p = 5.0, K_i = 0.5, K_d = 1.0 for the longitudinal PID controller and K_p = 0.75, K_i = 0.75, K_d = 0.3 for the lateral PID controller. The weights for different loss terms are as follows: $\lambda_F = 0.05$, $\lambda_{traj} = 1$, $\lambda_{ctl} = 1$, $\lambda_{aux} = 0.05$, and 0.001 for speed and value regression

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Table 1: Detailed statistics of the number of samples, the number of dynamic agents added, and a brief description of each town.

Town Name	#Samples	#Dynamic Agents	Description
Town01	50384	120	a basic town with T junction
Town02	55943	100	similar to Town01 but smaller
Town03	42771	120	a complex town
Town04	47954	200	a highway loop and a small town
Town05	53684	120	a squared-grid town with multiple lanes
Town06	48415	150	long highways
Town07	51549	110	a rural enviroment with narrow roads
Town10	59898	120	a city with various environments

respectively. For all experiments, we train TCP on 4 GeForce RTX 3090 GPUs. We use the Adam optimizer [10] with a learning rate of 1×10^{-4} and weight decay of 1×10^{-7} for all experiments. We train all models with batch size 128 for 60 epochs, and the learning rate is reduced by a factor of 2 after 30 epochs.

In the situation based fusion scheme, we choose whether the vehicle is turning as the criterion of the $situation$. Specifically, we calculate the absolute values of steer actions within the past 1 second. If half of them are larger than 0.1, we assume the vehicle is turning so the $situation$ is $control\ specialized$, otherwise $trajectory\ specialized$. For the online CARLA Leaderboard [1] submission, we use an asymmetric fusion scheme. If the $situation$ is $trajectory\ specialized$, we set $\alpha = 0.5$, and $\alpha = 0$ when it is $control\ specialized$. We take the maximum of the $brake$ control instead of taking the average. For the ensemble submission TCP-Ens, we also take the maximum of $brake$ value from different models and take the average for $steer$ and $throttle$.

C Experiments

C.1 Validation Protocol Details

We use the same validation routes as LAV [3]. This includes 4 routes in total, 2 from Town02 and 05 each. Each route is tested under 4 different weathers (ClearNoon, CloudySunset, SoftRainDawn, HardRainNight) and is repeated for 3 times, resulting in 48 routes in total. Random scenarios are added from the official CARLA leaderboard repo (all_towns_traffic_scenarios_public.json). The time-limit for agent blocking is reduced from 300 seconds to 60 seconds to save time.

C.2 Detailed Infractions Statistics

In this part, we report detailed infraction statistics of the methods on CARLA Leaderboard in Table 2, and statistics of our ablation experiments in Table 3 and Table 4.

C.3 Qualitative Results

We show cases of our method performing well in different challenging scenarios in Fig. 1. In the first case, the autonomous agent successfully reacts to the changing of the traffic light in time at the crossing. In the second case, a cyclist suddenly runs across the road right after the ego vehicle has made a right turn, and our agent makes an emergency brake in time, avoiding a collision. In the third case, the ego vehicle is making a right turn while there are other vehicles crossing. It stops and waits for the crossing vehicles to pass and then continues to make the turn. In the last case, our agent is performing an unprotected left turn with oncoming traffic, and it successfully negotiates with the oncoming vehicle.

More visualization examples of the trajectory-guided attention maps are provided in Fig. 2. We also show the GradCam [13] and EigenCam [11] visualization of two examples for Control-Only model with multi-step prediction scheme in Fig. 3. For the GradCam visualization, we set the target (which is needed to be maximized during the calculation of GradCam) be the negative action loss for the current and future action prediction. Note that GradCam visualizes the regions of the input image that are **important** for predictions by calculating gradients to maximize the target. It **does not** indicate that the model does focus or well capture the region highlighted by GradCam. As shown in Fig. 3, the

Table 2: Detailed statistics of the evaluation on the public CARLA Leaderboard [1] (accessed in May 2022). Driving Score, Route Completion, and Infraction Penalty are higher the better. For other metrics, lower values are desired. The collisions, infractions, and agent blocked related metrics are given as the number of events per kilometer. Our method outperforms other methods by a large margin in terms of Driving Score and Route Completion. We also have the best scores for metrics of collisions vehicle, collisions pedestrian, collisions layout, and off-road infractions among all methods.

Rank	Method	Driving Score	Route Completion	Infraction Penalty	Collisions Vehicle	Collisions Pedestrian	Collisions Layout	Red light Infractions	Off-road Infractions	Agent Blocked
1	**TCP-Ens** (ours)	**75.137**	85.629	**0.873**	0.316	**0.000**	**0.000**	0.089	0.038	0.537
1	**TCP** (ours)	69.714	82.962	0.851	**0.220**	0.006	0.034	0.083	**0.017**	0.564
1	**TCP-SB** (ours)	68.695	82.957	0.833	0.250	**0.000**	0.111	0.066	0.026	0.528
2	LAV [3]	61.846	**94.459**	0.640	0.696	0.038	0.017	0.166	0.252	**0.104**
3	Transfuser	61.181	86.694	0.714	0.814	0.036	0.007	**0.046**	0.228	0.428
4	Latent Transfuser	45.029	75.366	0.618	1.259	0.034	0.098	0.102	0.288	0.757
5	GRIAD [2]	36.787	61.855	0.597	2.772	**0.000**	0.407	0.484	1.388	0.842
6	Transfuser+ [9]	34.577	69.841	0.562	0.703	0.045	0.025	0.750	0.185	2.406
7	WoR [4]	31.370	57.647	0.557	1.346	0.606	1.017	0.791	0.963	0.473
8	MaRLn [14]	24.980	46.968	0.518	2.329	**0.000**	2.472	0.550	1.823	0.936
9	NEAT [5]	21.832	41.707	0.650	0.742	0.042	0.617	0.700	2.680	5.225

Table 3: Detailed infraction statistics of the ablation on the effectiveness of the trajectory-guided multi-step control prediction design.

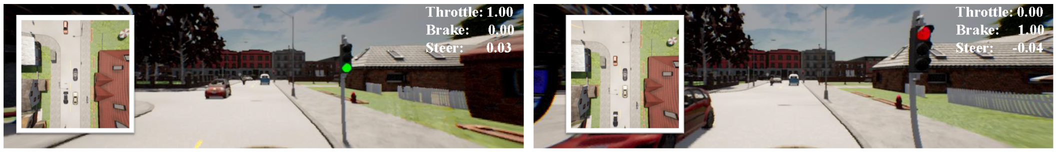
Exp.	Driving Score	Route Completion	Infraction Penalty	Collisions Vehicle	Collisions Pedestrian	Collisions Layout	Red light Infractions	Off-road Infraction	Agent Blocked
Control	32.45±2.23	76.54±3.22	0.45±0.03	1.24±0.06	0.00±0.00	0.23±0.09	0.18±0.05	0.59±0.06	0.41±0.11
+ traj-task	34.98±1.96	81.32±5.50	0.49±0.05	1.39±0.15	0.00±0.00	0.15±0.07	0.11±0.04	0.39±0.04	0.38±0.10
+ temporal	42.87±4.77	**87.51**±3.63	0.49±0.07	1.14±0.25	0.00±0.00	0.20±0.07	0.18±0.04	0.18±0.05	0.22±0.03
+ traj-attn	46.08±3.47	84.95±1.84	0.56±0.03	0.90±0.20	0.00±0.00	**0.04**±0.06	0.14±0.07	0.54±0.06	0.29±0.08
+ fusion	**57.01**±1.88	85.27±1.20	**0.67**±0.01	**0.37**±0.10	0.00±0.00	0.08±0.03	**0.10**±0.03	**0.14**±0.06	**0.20**±0.03

Table 4: Detailed infraction statistics of the experiments of the comparison between MTL and ensemble methods.

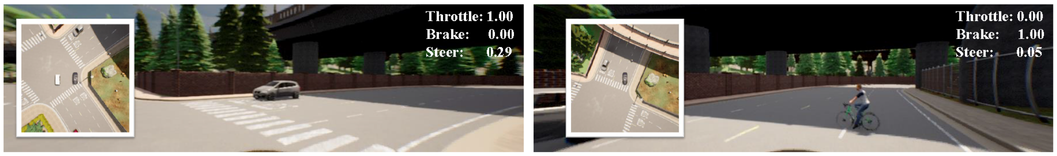
Exp.	Driving Score	Route Completion	Infraction Penalty	Collisions Vehicle	Collisions Pedestrian	Collisions Layout	Red light Infractions	Off-road Infraction	Agent Blocked
Ensemble	45.03±1.28	79.30±5.13	0.59±0.04	0.62±0.09	0.00±0.00	0.22±0.03	0.22±0.07	0.28±0.03	0.35±0.10
MTL	48.27±0.58	81.62±2.74	0.60±0.02	0.51±0.07	0.00±0.00	0.26±0.06	**0.06**±0.05	0.36±0.03	0.28±0.09
TCP-SB	52.46±4.66	83.94±3.75	0.64±0.04	0.53±0.14	0.00±0.00	0.08±0.03	0.13±0.09	**0.06**±0.00	0.29±0.04
TCP	57.01±1.88	85.27±1.20	0.67±0.01	**0.37**±0.10	0.00±0.00	0.08±0.03	0.10±0.03	0.14±0.06	**0.20**±0.03
TCP-Ens	**59.09**±3.66	**87.02**±2.02	**0.70**±0.03	0.41±0.19	0.00±0.00	**0.00**±0.00	0.10±0.13	0.18±0.05	0.27±0.06

GradCam heat-map for current action prediction ($\mathbf{a}_0$) focuses on regions close to the current location of the ego vehicle, while the heat-map for future prediction ($\mathbf{a}_1$) focuses on regions further. This indicates that predicting future actions does need to focus on further regions.

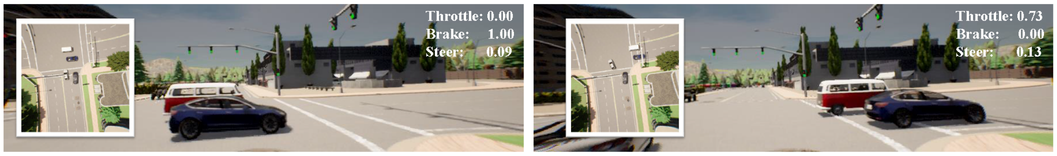
However, Control-Only model with multi-step action prediction only aggregates the image feature map once by global average. The pooled feature is then used to predict actions of all time steps. Therefore, it is **not** realistic to highlight corresponding important regions for each step. We use the EigenCam to visualize the 2D image feature map. It is a gradient-free visualization method to directly calculate the heat-map by projecting the feature map to eigen-vectors. As shown in the last column in Fig. 3, the highlighted region of the 2D image feature map only spans a single area, which is not informative enough for multi-step predictions. It verifies the necessity of re-aggregating the image information with different highlighted regions for each future step, as what we did in Fig. 2.



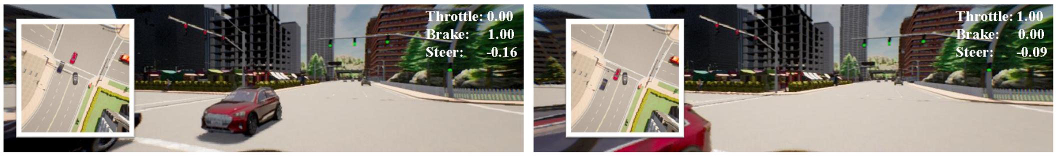
(a) Respond to traffic lights changes



(b) Emergency brake after a turning



(c) Right turn with crossing traffic



(d) Unprotected left turn with oncoming traffic

Figure 1: Examples of our agent performing well under different challenging scenarios.

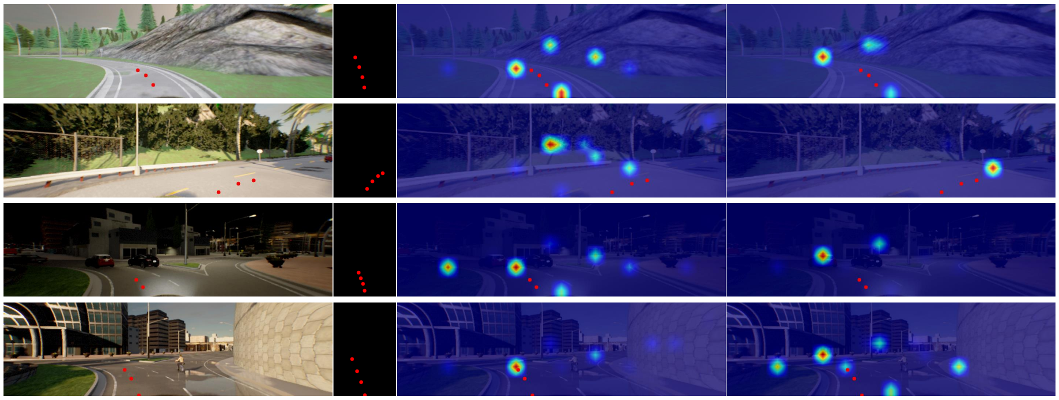


Figure 2: More examples of trajectory-guided attention maps. In each case (row), from the left to right we show that the input image with the predicted trajectory (the first waypoint is projected out of the image), the predicted trajectory in the top-down view, the attention map $\mathbf{w}_1$, the attention map $\mathbf{w}_3$.

D Discussion

D.1 Limitations and Future Work

D.1.1 Failure Cases and Future Work Directions

Our work mainly focuses on combining the two output forms of end-to-end autonomous driving, *i.e.*, trajectory planning and direct control. A detailed and elaborate situation based fusion scheme is based on rules which may require a large number of experiments and specific prior knowledge. A more general or a learning-based adaptive fusion scheme may be a possible future direction.

We also discuss two typical failure cases of TCP in Fig. 4. The first scenario happens when other vehicles initially outside the ego agent's front view rushes into the path with a high speed. It causes

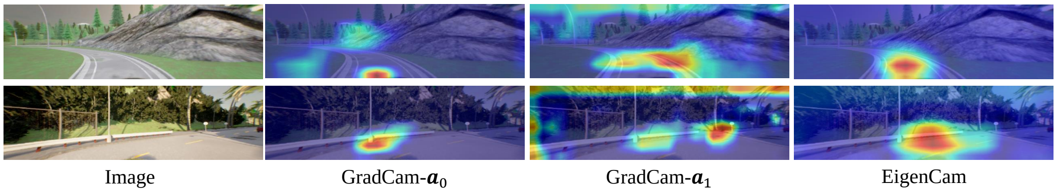


Figure 3: Visualization examples of GradCam [13] and EigenCam [11]. From left to right: original image, GradCam heat-map for current action prediction, GradCam heat-map for future action prediction, EigenCam heat-map for the image feature map.

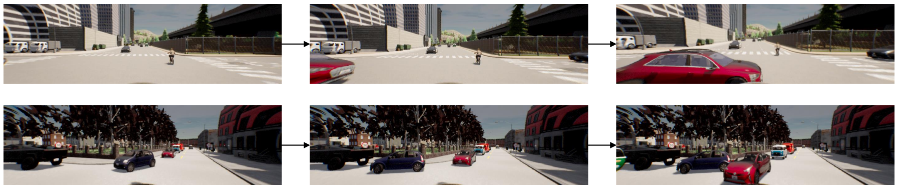


Figure 4: Examples of two failure cases. Top row: the red vehicle runs into the ego path with a high speed, and the ego vehicle fails to take an emergent brake. Down row: The ego agent is waiting for a left turn but occupies part of the opposite lane, causing a block.

a delayed collision when an emergent braking fails. It is because of the limited view of our single camera, hence a straightforward future direction is to add multi-view cameras or a LiDAR input to our agent. Another kind of failures is that the ego agent fails to predict the possible trajectory of other vehicles, resulting in blocking or collisions. Thus explicitly making trajectory predictions of other vehicles and combining it with our trajectory branch is also an interesting direction to further boost the ability of generalization as demonstrated in LAV [3] and LBW [15].

D.2 Broader Impact

We explore the limitations and advantages of the two conventional output paradigms for end-to-end autonomous driving, present TCP which achieves state-of-the-art performance on the public closed-loop benchmark to push the boundary of the problem. We aim to bring together the two branches of research in this field and provide a unified framework to combine their possible advantages. Our work provides a simple yet effective framework, based on which, new models and techniques can be conveniently integrated and transparently compared. Despite such improvement, we fully understand that our work is by no means perfect and still has many challenges when it comes to real-world application. Our model is trained and tested in the simulator, directly deploying it in the real world will lead to possible traffic accidents which may cause negative societal impacts.

E License of Assets

CARLA [7] is an open-source simulator which is under the MIT license and its assets are under the CC-BY license. We integrate part of the official code of Roach [16] which is under the CC-BY-NC 4.0 license into our codebase. The pretrained ResNet model is under the MIT license.

The source code and training data for our work will be publicly available once accepted and they are under the CC-BY-NC 4.0 license.

References

[1] CARLA autonomous driving leaderboard. `https://leaderboard.carla.org/`, 2022. 2, 3

[2] Raphael Chekroun, Marin Toromanoff, Sascha Hornauer, and Fabien Moutarde. Gri: General reinforced imitation and its application to vision-based autonomous driving. *arXiv preprint arXiv:2111.08575*, 2021. 3

[3] Dian Chen and Philipp Krähenbühl. Learning from all vehicles. In *CVPR*, 2022. 1, 2, 3, 5

Table 5: Detailed network structure of our TCP model.

Layer Type	# of Filters	Activation Function	#
Image Encoder			
ResNet-34			
Measurement Encoder			
FC	128	ReLU	× 2
Join_ctrl			
FC	512	ReLU	× 2
FC	256	ReLU	× 1
Join_traj			
FC	512	ReLU	× 2
FC	256	ReLU	× 1
Speed Head			
FC	256	ReLU	× 2
FC	1	ReLU	× 1
Value Head ×2 traj+ctrl			
FC	256	ReLU	× 2
FC	1	ReLU	× 1
Temporal Module			
GRU_cell	hidden size = 256		× 1
FC (output)	256	ReLU	× 2
Control Policy Head			
FC	256	ReLU	× 2
FC (alpha)	2	Softplus	× 1
FC (beta)	2	Softplus	× 1
Traj Policy Head			
GRU_cell	hidden size = 256		× 1
FC (output)	2		× 2
Init Att.			
FC	256	ReLU	× 1
FC	29*8	Softmax	× 1
Traj Guided Att.			
FC	256	ReLU	× 1
FC	29*8	Softmax	× 1
Merge (merge the re-aggregated image feature and hidden state)			
FC	512	ReLU	× 1
FC	256	ReLU	× 1

[4] Dian Chen, Vladlen Koltun, and Philipp Krähenbühl. Learning to drive from a world on rails. In *ICCV*, 2021. 3

[5] Kashyap Chitta, Aditya Prakash, and Andreas Geiger. Neat: Neural attention fields for end-to-end autonomous driving. In *ICCV*, 2021. 1, 3

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1

[7] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *CoRL*, 2017. 5

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[9] Bernhard Jaeger. Expert drivers for autonomous driving. Master's thesis, University of Tübingen, 2021. 3

[10] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 2

[11] Mohammed Bany Muhammad and Mohammed Yeasin. Eigen-cam: Class activation map using principal components. In *IJCNN*, 2020. 2, 5

[12] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *CVPR*, 2021. 1

[13] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017. 2, 5

[14] Marin Toromanoff, Emilie Wirbel, and Fabien Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *CVPR*, 2020. 3

[15] Jimuyang Zhang and Eshed Ohn-Bar. Learning by watching. In *CVPR*, 2021. 5

[16] Zhejun Zhang, Alexander Liniger, Dengxin Dai, Fisher Yu, and Luc Van Gool. End-to-end urban driving by imitating a reinforcement learning coach. In *ICCV*, 2021. 1, 5