Probabilistic End-to-End Vehicle Navigation in Complex Dynamic Environments With Multimodal Sensor Fusion

Peide Cai, Sukai Wang, Yuxiang Sun, and Ming Liu, *Senior Member, IEEE*

***Abstract*—All-day and all-weather navigation is a critical capability for autonomous driving, which requires proper reaction to varied environmental conditions and complex agent behaviors. Recently, with the rise of deep learning, end-to-end control for autonomous vehicles has been well studied. However, most works are solely based on visual information, which can be degraded by challenging illumination conditions such as dim light or total darkness. In addition, they usually generate and apply deterministic control commands without considering the uncertainties in the future. In this letter, based on imitation learning, we propose a probabilistic driving model with multi-perception capability utilizing the information from the camera, lidar and radar. We further evaluate its driving performance online on our new driving benchmark, which includes various environmental conditions (e.g., urban and rural areas, traffic densities, weather and times of the day) and dynamic obstacles (e.g., vehicles, pedestrians, motorcyclists and bicyclists). The results suggest that our proposed model outperforms baselines and achieves excellent generalization performance in unseen environments with heavy traffic and extreme weather.**

***Index Terms*—Automation technologies for smart cities, autonomous vehicle navigation, multi-modal perception, sensorimotor learning, motion planning and control.**

I. Introduction

In the field of autonomous driving, traditional navigation methods are commonly implemented with modular pipelines [1], [2], which split the navigation task into individual sub-problems, such as perception, planning and control. These modules often rely on a multitude of engineering components to produce reliable environmental representations, robust decisions and safe control actions. However, since the separate modules rely on each other, the system can lead to an accumulation of errors. Therefore, each component requires careful and time-consuming hand engineering.

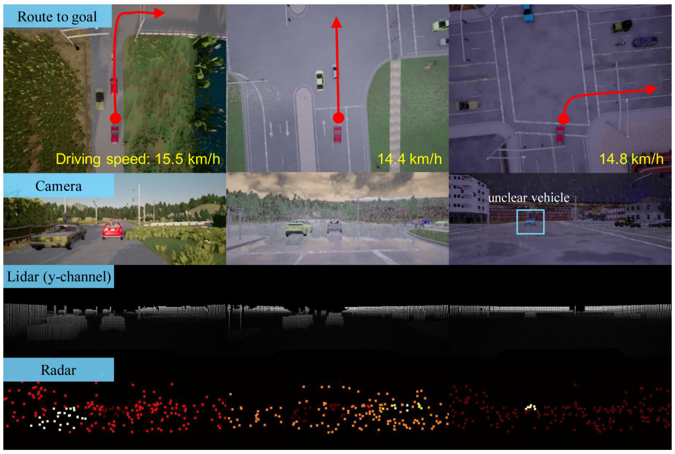


Fig. 1. Snapshots of different driving scenarios (left to right: *ClearDay*, *RainySunset* and *DrizzleNight*) with global route directions and sensor data information. For visualization, we project the lidar data (y-channel, i.e., the height information) and radar data (relative speed to the ego-vehicle) to the image plane. Brighter points mean larger values. It can be seen that the information characteristic from lidar and radar is more consistent than from the camera in different environmental conditions.

In recent years, with the unprecedented success of deep learning, an alternative method called end-to-end control [3]–[12] has arisen. This paradigm mimics the human brain and maps the raw sensory input (e.g., RGB images) to control output (e.g., steering angle) in an end-to-end fashion. In addition, it substitutes laborious hand engineering by learning control policies directly on data from human drivers with deep networks, where explicit programming or modeling of each possible scenario is not needed. Moreover, it can adapt to complex noise characteristics of different environments during training, which cannot be captured well by analytical methods.

While end-to-end driving has been considerably fruitful, there exist three critical deficiencies in the prior works.

1) The visual information is stressed too much. Most works depend solely on cameras for scene understanding and decision making [3]–[14]. However, although cameras are versatile and cheap, they have difficulty capturing fine-grained 3-D information. In addition, perception relying on cameras is prone to be affected by challenging illumination and weather conditions, such as the *DrizzleNight* case shown in Fig. 1. Because of dim light and rain drops in this scene, the blue car far ahead left

Manuscript received February 24, 2020; accepted April 21, 2020. Date of publication May 11, 2020; date of current version May 22, 2020. This letter was recommended for publication by Associate Editor H. Myung and Editor Y. Choi upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China, under Grant U1713211, in part by the Shenzhen Science, Technology and Innovation Commission (SZSTI) under Grant JCYJ20160428154842603, and in part by the Research Grant Council of Hong Kong SAR Government, China, under Project Nos. 11210017 and 21202816, awarded to Prof. Ming Liu. *(Corresponding author: Ming Liu.)*

The authors are with The Hong Kong University of Science and Technology, Hong Kong (e-mail: pcaiaa@connect.ust.hk; swangcy@connect.ust.hk; sun.yuxiang@outlook.com; eelium@ust.hk).

This letter has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/LRA.2020.2994027

can be difficult to recognize. In such scenarios, vision-based driving systems can be dangerous. However, the blue car is quite distinguishable by observing the speed distribution from the radar data.

2) The probabilistic nature of executable actions is not well explored. Most works output deterministic commands to the vehicle [15], [16]; however, non-determinism is a key aspect of controlling, which is useful in many safety-critical tasks such as collision checking and risk-aware motion planning [17]. A more reasonable approach, therefore, should be predicting a motion distribution indicating *what could do* rather than *what to do* for the driving platform.

3) The prior end-to-end methods are not evaluated sufficiently in terms of the *navigation* task. Most works are evaluated by first collecting a driving dataset with ground-truth annotations (e.g., expert control actions) and then measuring the average prediction error *offline* on the test set [6], [9], [10], [13], [14], [17]. However, different from the computer vision tasks such as object detection, the priority of driving should be safety and robustness rather than accuracy. As indicated in [18], the offline prediction error cannot well reflect the actual driving quality. Therefore, *online* evaluation is more reasonable and should be given more attention. One critical concern for online evaluation is the environmental complexity, yet prior related works either test their methods in static maps [11], [12], [16], [19], [20], or scenarios with low-level complexity [3]–[5], [7], [8], [15].

The aforementioned limitations motivate our exploration to enhance the perception capability for end-to-end driving systems. To this end, we propose a mixed sensor setup combining a camera, lidar and radar. The multimodal information is processed by uniform alignment and projection onto the image plane. Then, ResNet [21] is used for feature extraction. Based on this setup, we introduce a probabilistic motion planning (PMP) network to learn a deep probabilistic driving policy from expert provided data, which outputs both a distribution of future motion based on the Gaussian mixture model (GMM) [9], [17], [22], and a deterministic control action. Finally, we evaluate the driving performance of our model online on a new benchmark with extensive experiments. The main contributions of this letter are summarized as follows.

- An end-to-end navigation method with multimodal sensor fusion and probabilistic motion planning, named PMP-net, for improving perception capability and considering uncertainties in the future.
- A new online benchmark, named *DeepTest*, to perform analysis of driving systems in high-fidelity simulated environments with varied maps, weather, lighting conditions and traffic densities.
- Extensive evaluation and human-level driving performance of the proposed PMP-net, presented in unseen urban and rural areas with extreme weather and heavy traffic.

II. Related Work

End-to-end control is designed with deep networks to directly learn a mapping from raw sensory data to control outputs. The pioneer ALVINN system [23] developed in 1989 uses a multilayer perceptron to learn the directions a vehicle should steer. With the recent advancement of deep learning, end-to-end control techniques have experienced tremendous success. For example, using more powerful modern convolutional neural networks (CNNs) and higher computational power, Bojarski *et al.* [3] demonstrate impressive performance in simple real-world driving scenarios such as on flat or barrier-free roads. Xu *et al.* [6] develop an end-to-end architecture to predict future vehicle egomotion from a large-scale video dataset. However, these works only realize a lane-following task and goal-directed navigation is not studied.

To enable goal-directed autonomous driving, Codevilla *et al.* [5] propose a conditional imitation learning pipeline. In this work, the vehicle is able to take a specific turn at intersections based on high-level navigational commands such as *turn left* and *turn right*. Follow-up works include [7], [12], [13] and [14]. Another trend of adding guidance to the control policy is using global route, which is a richer representation of the intended moving directions than turning commands. For example, Gao *et al.* [4] render routes on 2D floor maps and call them *intentions*. Then, a neural-network motion controller maps *intentions* and camera images directly to robot actions. Pokle *et al.* [16] follow this idea and implement a deep local trajectory planner and a velocity controller to compute motion commands based on the path generated by a global planner. However, these two works only focus on indoor robot navigation. For outdoor driving applications, Cai *et al.* [20] realize high-speed autonomous drifting in racing scenarios guided by route information with deep reinforcement learning. However, the control policy is only evaluated in static maps. Hecker *et al.* [10] propose to learn a control policy with GPS-based route planners and surround-view cameras. However, as with many other works [6], [9], [13], [17], this work is only evaluated offline by analysing the average predicting error, providing unclear information of the actual driving quality.

Inspired by the route-guided navigation methods mentioned above, we use a global planner to compute paths towards destinations in outdoor driving areas. For the low-level reactive control, we implement an end-to-end network translating the global route into driving actions (steering, throttle and brake). Based on this architecture, point-to-point autonomous driving can be realized. The network is trained with imitation learning and can adapt to varied environments to drive appropriately (e.g., slow down at intersections) and safely (e.g., slow down for a car, and urgently stop for jaywalkers). Similar to [4] and [16], we assume that the localization information is available during system operation. However, different to [4] and [16], our work focuses on complicated outdoor driving scenarios, and combines multimodal sensors complementing each other to generate unified perception results.

In addition, our approach relates to the work of probabilistic driving models. To improve the capability of handling long-term plans with imitation learning, Amini *et al.* [9] propose a variational network to predict a full distribution over possible steering commands. Similarly, Huang *et al.* [17] propose to use GMM to predict a distribution of future vehicle trajectories. These works explicitly consider uncertainties of future motions on logged data with **offline** metrics. By contrast, we evaluate our probabilistic driving model **online** with varied environmental

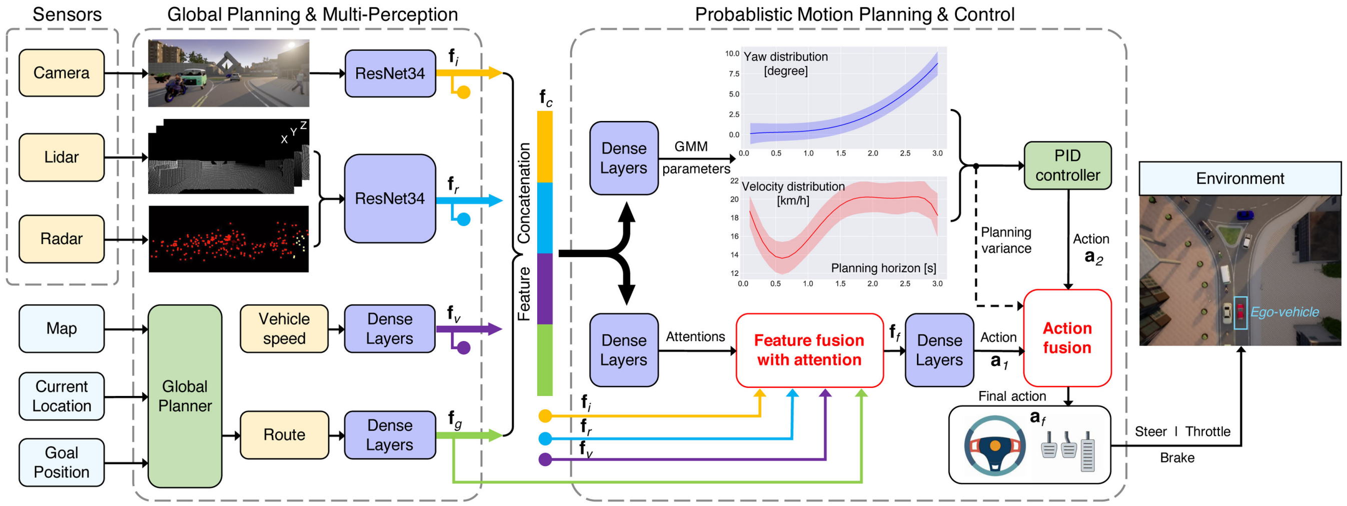


Fig. 2. The architecture of our probabilistic motion planning network (PMP-net). It receives the multimodal sensory input and plans a motion distribution for 3 seconds in the future, based on which a PID controller is designed to generate a control action a_2. In addition, PMP-net generates another action a_1 in an end-to-end fashion. Then the variance of the planned motion distribution is used to fuse the dual actions for controlling the vehicle.

conditions (e.g., rainy night with heavy traffics), which has not been studied in this context before.

III. Methodology

A. Formulation

We formulate the problem of autonomous vehicle navigation as a goal-directed motion planning task to be solved by an end-to-end network architecture with imitation learning. The goal is to control the vehicle to drive safely and robustly in complex outdoor areas to achieve point-to-point navigation, like a human driver. To this end, we design a probabilistic driving model using multimodal perceptions from the camera, lidar and radar. In addition, we choose the latest CARLA simulation (0.9.7) [24] to train and evaluate the system.[1] The entire pipeline of our PMP-net is shown in Fig. 2.

B. Dataset Collection

To make the model successfully learn the knowledge of goal-directed reactive control in the context of outdoor driving, we collect a large-scale dataset with a global planner and an expert demonstrator in CARLA. At the beginning of each driving episode, the ego-vehicle is spawned at a random position $\boldsymbol{p}$. Then a collision-free coarse route (ranging from 350 m to 1500 m) from $\boldsymbol{p}$ to a destination $\boldsymbol{d}$ is provided by a global planner. The vehicle then follows this route at a speed of around 40 km/h while reacting to local environments to avoid collisions, such as slowing down for a forward-facing car that is moving slowly. Additionally, the vehicle reasonably slows the speed down to 15 km/h at intersections to ensure safety. In the process of data collection, we record the vehicle velocities, yaw angles, RGB images, lidar/radar data and expert driving actions (i.e., steering, throttle and brake) at 10 Hz. Moreover, in order to increase the complexity of our dataset, we focus on the following two aspects:

1) Complexity of Environments: a) The datasets from prior works [5], [7], [8] are generated only in one map with two lanes and 90-degree turns (*Town01* in Fig. 3). By contrast, we use five urban maps for data collection, which consist of different types of intersections and even roundabouts, and multiple lanes on roads; b) We set nine combinations of weather (*clear*, *drizzle* and *rainy*) and illumination (*daytime*, *sunset* and *night*). Heavier rain leads to more puddles on roads, and thus brings a greater reflection effect for visual perception.

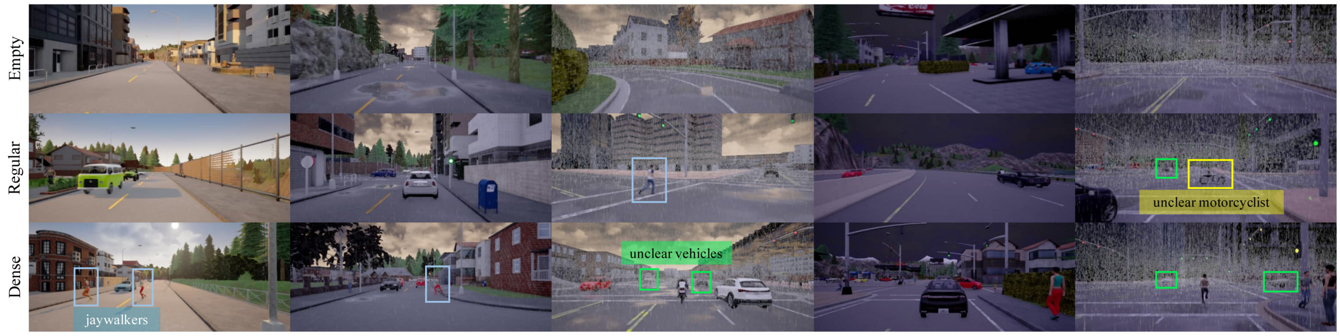
2) Complexity of Road Agents: a) We set pedestrians with different appearances (children and adults) randomly running or walking along the sidewalks and crosswalks. They occasionally disobey traffic rules and cross the road abruptly without previous notice, which increases the safety burden for autonomous driving; b) We set different types of vehicles (e.g., cars, trucks, vans, jeeps, buses, motorcyclists and bicyclists) with multiple appearances navigating around the cities at varied speeds. Based on a) and b), we apply four levels of traffic density for data collection: *empty*, *few*, *regular* and *dense*. Note that these road agents are controlled by the AI engine from CARLA to construct realistic city scenarios.

The setups mentioned above can be partially viewed in Fig. 3 and more can be viewed in our supplementary videos. These help to generate sufficient interactions between the ego-vehicle and road agents in diverse environments. Based on these setups, we finally collect 360 high-fidelity driving episodes, which last 10.8 hours in total with 389 thousand frames and cover a driving distance of 247 km.

C. Model Architecture

1) Global Planning: The global planner is separate from the deep networks. It is implemented with the A^* algorithm to plan a high-level coarse route from the start point to the destination

[1] Different from the older versions of CARLA (0.8.x) used in [5], [7] and [8], which contain only two urban maps, the latest CARLA environment provides seven maps covering both urban and rural areas, with more available sensors, improved physical dynamics and more realistic illuminations. http://carla.org/2019/12/11/release-0.9.7/



(a) Town01-ClearDay (b) Town02-DrizzleSunset (c) Town03-RainySunset (d) Town04-ClearNight (f) Town05-RainyNight

Fig. 3. Overview of our dataset: varied maps, weather and illumination conditions with increasing traffic densities (top to bottom). Noticeable road agents are bounded by color boxes. Note that this figure shows only a small part of the environmental setups; please see contexts in Section III-B for more details. Columns (a–c) show there can sometimes be jaywalkers running across the roads, for which the ego-vehicle will urgently slow down or completely stop to ensure safety. In addition, it can be seen that in rainy scenarios, especially in *RainyNight*, the surroundings are considerably blurred (e.g., the unclear motorcyclist in the *Regular* setting of column (f)), leading to potential risks for the vision-based driving models [5], [7], [8], [14].

based on static town maps. Similar to [16] and [20], we downsample the full global route $\boldsymbol{G}_f$ to local relevant routes $\boldsymbol{G}$ during navigation, which is shown in (1):

$$\boldsymbol{G} = \{(x_k, y_k)\,|1 \leq k \leq 130\} \subset \boldsymbol{G}_f. \tag{1}$$

Note that the first waypoint (x_1, y_1) in $\boldsymbol{G}$ is the closest waypoint in $\boldsymbol{G}_f$ to the current location of the vehicle, and the distance of every two adjacent points is 0.4 m. The waypoints are then flattened into a 260-dimensional vector to be processed by dense layers with fully connected ReLU layers. The extracted feature is a higher dimensional vector $\boldsymbol{f}_g \in \mathbb{R}^{2048}$.

2) Multi-Perception: With the aim to capture environmental information, the camera records color textures in a 2D image plane, while the lidar captures 3-D spatial locations and the radar records movement information (i.e., speeds of obstacles relative to the ego-vehicle). We combine these sensors together in our network so that the vehicle is able to sense different dimensions of its surroundings.

Specifically, we project the lidar point clouds and radar data to the image plane with the same width and height as the camera images. We name it the *ralidar* image (250 × 600 × 4), in which the first three channels encode 3-D coordinates and the forth channel encodes relative speeds, as shown in Fig. 4. In this way, the multimodal measurements are aligned on the same space and can be uniformly processed with CNNs. In this work, we use ResNet34 [21] as the backbone to extract environmental features from the camera and ralidar images. The results are feature vectors $\boldsymbol{f}_i \in \mathbb{R}^{2048}$ and $\boldsymbol{f}_r \in \mathbb{R}^{2048}$.

3) End-to-End Action Generation: In addition to the sensory data and the global route, our network also takes as input the velocity of the ego-vehicle (v_x, v_y) to the dense layers. The extracted feature is a higher dimensional vector $\boldsymbol{f}_v \in \mathbb{R}^{2048}$. Then the features $[\boldsymbol{f}_i, \boldsymbol{f}_r, \boldsymbol{f}_v, \boldsymbol{f}_g]$ are handled in two ways: a) we concatenate them into a vector $\boldsymbol{f}_c \in \mathbb{R}^{8192}$ for further processing, and b) in the spirit of [16], we fuse them with an attention mechanism defined in (2), where the coefficients $\boldsymbol{a} = [a_i, a_r, a_v, a_g]$ reflect the relative importance of the features in changing environments.

$$\boldsymbol{f}_f = a_i\boldsymbol{f}_i + a_r\boldsymbol{f}_r + a_v\boldsymbol{f}_v + a_g\boldsymbol{f}_g. \tag{2}$$

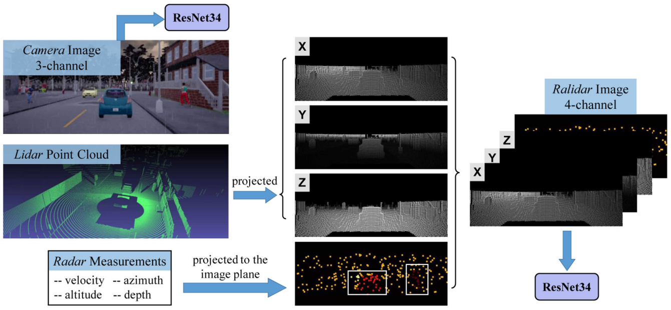


Fig. 4. Multimodal data processing. We achieve data alignment by projecting the lidar pointclouds and radar measurements to the image plane by combining them together to form the *ralidar* image. Then, two ResNet34 modules are used to extract features from the camera and ralidar images. Brighter points mean larger values in the projected images. Noticeable road agents in the projected radar image are bounded by white boxes.

The coefficients $\boldsymbol{a}$ are computed by transforming $\boldsymbol{f}_c$ with dense layers and softmax activation. After such feature fusion, a control action $\boldsymbol{a}_1$ composed of steering, throttle and brake is generated by projecting $\boldsymbol{f}_f$ with fully connected ReLU layers. Inspired by [18], we use the L1 loss function for this module as it is better correlated to the online driving performance.

4) Probabilistic Motion Planning: In this module, we aim to learn a full parameterized distribution over possible ego-motions (i.e., velocities and yaw angles) for 3.0 s into the future, as shown in Fig. 2. We adopt the GMM to represent such a distribution due to its excellent approximation properties. Specifically, the combined feature $\boldsymbol{f}_c$ in our work is transformed by dense layers into GMM parameters (i.e., weight, mean and variance) to describe the distribution of future motions. Similar to [9] and [17], the negative log-likelihood (NLL) loss function is used for this module.

As mentioned in [22], the advantage of probabilistic modeling is that we can make a decision by evaluating its statistical properties. In this work, based on the mean values $(\boldsymbol{\mu})$ of the planned motion distribution, we further design a PID controller to calculate a control action $\boldsymbol{a}_2$ composed of steering, throttle and brake. The target point for this PID controller (assume k

TABLE I
WE EVALUATE DIFFERENT DRIVING MODELS ON OUR *DEEPTEST* DRIVING BENCHMARK. ↑ MEANS LARGER NUMBERS ARE BETTER, ↓ MEANS SMALLER NUMBERS ARE BETTER. THE BOLD FONT HIGHLIGHTS THE BEST RESULTS IN EACH COLUMN

	Training Conditions			New Weather			New Town			New Town & Weather		
Town Name	`Town03` (urban)			`Town05` (urban)			`Town07` (rural)			`Town06` (urban)		
Traffic Density	Empty	Regular	Dense	Empty	Regular	Dense	Empty	Regular	Dense	Empty	Regular	Dense
Success Rate ↑ (%)												
CIL [5]	38	16	16	33	11	0	0	0	0	16	11	0
CIL-R	83	55	38	33	22	16	22	11	11	11	11	11
INT [4]	16	33	11	83	5	5	38	22	5	94	61	16
PMP (*ours*)	**100**	**72**	**88**	**100**	**77**	**77**	**100**	**83**	**72**	**100**	**88**	**83**
Wrong Lane ↓ (%)												
CIL [5]	66.05	45.16	50.87	57.22	64.41	46.18	35.55	36.81	40.71	44.14	52.37	52.03
CIL-R	26.60	25.57	19.07	26.58	36.64	41.86	8.88	7.20	3.35	42.50	50.72	51.61
INT [4]	**0.00**	0.04	**0.01**	**0.07**	**0.12**	**0.15**	**0.00**	**0.00**	**0.00**	**0.12**	**0.13**	**0.28**
PMP (*ours*)	0.02	**0.00**	**0.01**	0.40	0.48	0.50	0.04	**0.00**	0.01	0.43	0.40	0.61
Overspeed ↓ (%)												
CIL [5]	0.33	0.37	0.16	0.10	**0.00**	–	–	–	–	**0.04**	**0.00**	–
CIL-R	0.14	**0.13**	**0.08**	**0.04**	**0.00**	**0.00**	0.33	**0.09**	**0.28**	0.10	0.16	1.54
INT [4]	17.70	11.18	5.85	17.09	15.14	8.52	19.03	11.87	14.84	37.12	30.22	31.04
PMP (*ours*)	**0.11**	0.22	0.12	0.14	**0.00**	0.06	**0.26**	0.30	**0.28**	0.40	0.28	**0.36**

frames in the future) is set to the point 5 m ahead of the vehicle by calculating the integral with $\boldsymbol{\mu}$. Then, the final action $\boldsymbol{a}_f$ to control the vehicle is computed by examining the reliability of the motion distribution through its accumulated variance $\boldsymbol{\sigma}^2$:

$$\boldsymbol{a}_f = (1-\lambda)\,\boldsymbol{a}_1 + \lambda\boldsymbol{a}_2,\ \lambda = e^{-c_1\cdot\max\left(0,\ \sum_i^k \sigma^2 - c_2\right)}. \tag{3}$$

In this way, higher planning uncertainty leads to smaller λ, thus the final action will depend more on $\boldsymbol{a}_1$. We believe that we can take advantage of both end-to-end control and probabilistic modeling by performing such reliability-aware action fusion.

IV. EXPERIMENTS AND DISCUSSION

A. Training Setup

We train the proposed PMP-net on our large-scale driving dataset introduced in Section III-B. The full dataset is divided into a training set and a validation set according to the ratio of 7:1, leading to 340 K training samples.[2] We use the Adam optimizer with a learning rate of 0.0001, and the batch size is 90. Based on these setups, the model is trained on two Nvidia GeForce RTX 2080 Ti GPUs for about 75 hours, with 234 K training steps to achieve convergence. For comparison, we also train and finetune three other baselines on the same training set, which are for visual navigation:

- **CIL**: The conditional imitation learning network introduced in [5]. This maps the camera images and ego-velocities directly to control actions, based on four discrete commands for goal-directed navigation: *follow lane*, *turn left*, *turn right* and *go straight at the intersection*.
- **CIL-R**: We replace the original image processing module of CIL (which is relatively shallow) with ResNet34, to evaluate if deeper models perform better for our task.
- **INT**: The intention-net introduced in [4] with the backbone of ResNet34 for fair comparisons. This maps the camera images and global routes to control actions. Note that the original intention-net takes the indoor floor maps rendered with routes for directions. We replace it with the local relevant routes $\boldsymbol{G}$ introduced in (1).

B. Evaluation

1) DeepTest Benchmark: We evaluate the online driving performance for different models on our proposed *DeepTest* benchmark in CARLA. Compared with the previous benchmarks in [7] and [24], *DeepTest* has many more environmental setups, such as more test maps, weather conditions and interactions with road agents. In addition, different to [7] and [24], we set zero tolerance for collision events, which means that any degree of collisions with static (e.g., trees) or dynamic (e.g., pedestrians) objects leads to a failed episode.

In our benchmark, different methods are tested on four maps. For each map, we set three levels of traffic densities: *empty*, *regular* and *dense*. Therefore, each driving model relates to 12 driving tasks. Note that denser traffic leads to harder driving tasks as it involves more dynamic obstacles on the road. In each task, we further set 18 goal-directed episodes with varied weather conditions. Therefore, to fully evaluate PMP-net and the other three baselines, 864 driving episodes should be conducted. Finally, the evaluation process costs 4 days on our computer and covers a driving distance of 855 km. Compared with the environmental setups in the training set (Section III-B), we consider new maps, illuminations and weather in *DeepTest* to test the generalization capability. Specifically, we add an unseen

[2] Note the test set is not considered because we evaluate our model *online* in Section IV-B by making the ego-vehicle directly interact with dynamic environments.

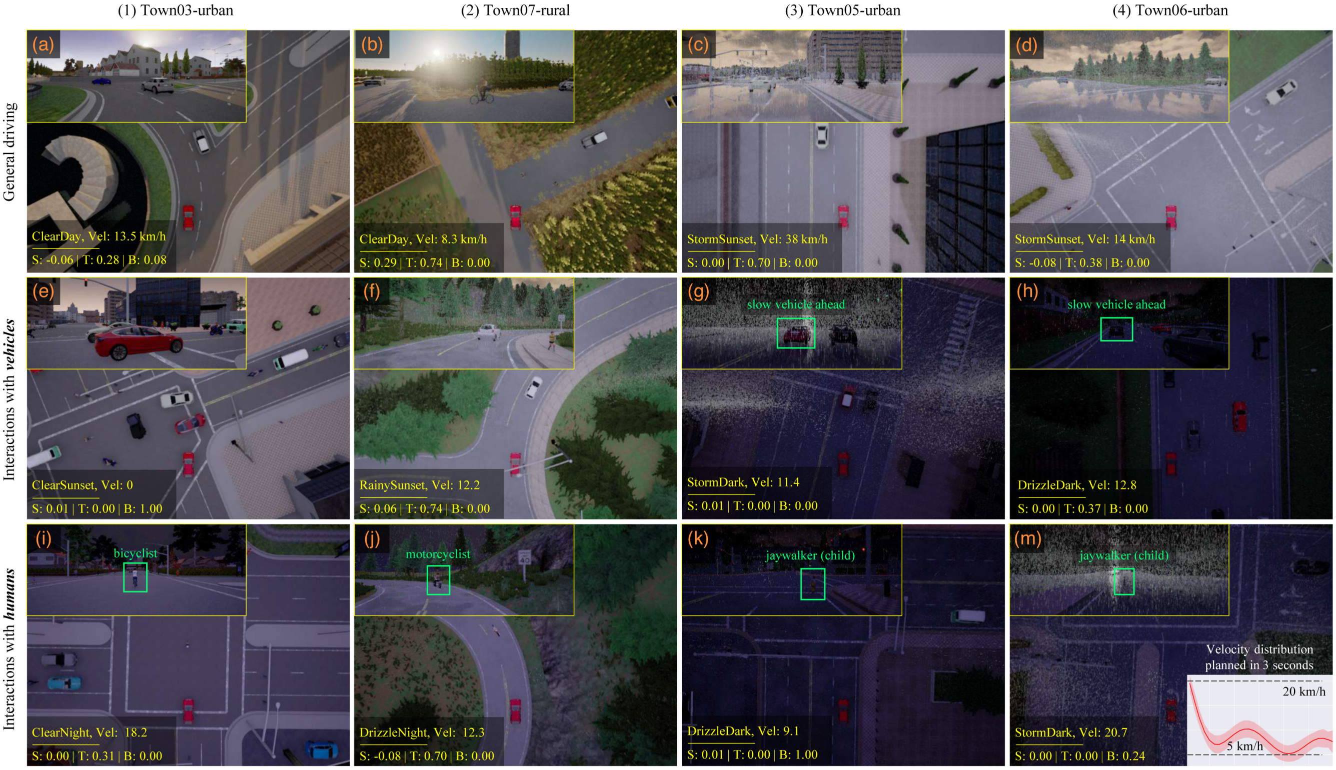

(1) Town03-urban
(2) Town07-rural
(3) Town05-urban
(4) Town06-urban
General driving
Interactions with *vehicles*
Interactions with *humans*
(a)
ClearDay, Vel: 13.5 km/h
S: -0.06 | T: 0.28 | B: 0.08
(b)
ClearDay, Vel: 8.3 km/h
S: 0.29 | T: 0.74 | B: 0.00
(c)
StormSunset, Vel: 38 km/h
S: 0.00 | T: 0.70 | B: 0.00
(d)
StormSunset, Vel: 14 km/h
S: -0.08 | T: 0.38 | B: 0.00
(e)
ClearSunset, Vel: 0
S: 0.01 | T: 0.00 | B: 1.00
(f)
RainySunset, Vel: 12.2
S: 0.06 | T: 0.74 | B: 0.00
(g)
slow vehicle ahead
StormDark, Vel: 11.4
S: 0.01 | T: 0.00 | B: 0.00
(h)
slow vehicle ahead
DrizzleDark, Vel: 12.8
S: 0.00 | T: 0.37 | B: 0.00
(i)
bicyclist
ClearNight, Vel: 18.2
S: 0.00 | T: 0.31 | B: 0.00
(j)
motorcyclist
DrizzleNight, Vel: 12.3
S: -0.08 | T: 0.70 | B: 0.00
(k)
jaywalker (child)
DrizzleDark, Vel: 9.1
S: 0.01 | T: 0.00 | B: 1.00
(m)
jaywalker (child)
Velocity distribution planned in 3 seconds
20 km/h
5 km/h
StormDark, Vel: 20.7
S: 0.00 | T: 0.00 | B: 0.24

Fig. 5. Online evaluation results of PMP-net in our *DeepTest* benchmark. The environment setups, driving velocities and control actions are shown in yellow text. Noticeable road agents (e.g., jaywalkers) are bounded by green boxes. The range of steering is $[-1, 1]$, while for throttle and brake the range is $[0, 1]$. The sample driving behaviors are: (c) lane-following, turning at (b, d, e) intersections or (a) roundabouts, (g) lane-changing, (f, h, i, j) vehicle-, bicyclist- or motorcyclist-following, and (k, m) urgently slowing down for jaywalkers. All of these behaviors are performed autonomously and safely by PMP-net in an end-to-end fashion without hand-crafted rules.

rural map *Town07* and an urban map *Town06*. *Town07* brings new challenges to test the negotiation skills with narrow roads and many non-signalized crossings. In addition, we add four extreme illumination and weather conditions: *ClearDark*, *DrizzleDark*, *StormDark* and *StormSunset*. The new *Dark* and *Storm* (i.e., heavy rain) settings, which are shown in Fig. 5, bring extra challenges to the drive with limited vision. Similar to [5], we do not consider traffic lights in this work. For quantification of the driving performance, three metrics are adopted as follows:

- **SR**: success rate. An episode is considered to be successful if the agent reaches a certain goal without any collision within a time limit. Based on this, we calculate the success rate for models in different tasks.
- **WL**: The proportion of the period in a wrong lane to the total driving time.
- **OVSP**: The proportion of the overspeeding period to the total driving time. The speed limit is set to 20 km/h at intersections and 50 km/h elsewhere.

2) Quantitative Analysis: We show the results on the *DeepTest* benchmark in Table I. In the following, the analyses are given from two perspectives: the *ability* and the *quality* of autonomous driving.

Ability: Success rate (SR) is used to measure the self-driving ability, which is a crucial concern in this area.

It can be seen that the CIL model presents the worst results, which can not even achieve a successful episode in `Town07`. In addition, although in `Town03` we only set new routes with similar environments to the training dataset, CIL still presents low SRs (16∼38%). With the help of a deeper backbone in CIL-R, the performance is improved. For example, the SR in `Town03-empty` increases from 38% to 83%.

By changing the model structure to INT, better generalization performance on certain new environments is achieved, for example, the SR in `Town06-Regular` increases from 11% to 61%. However, INT performs worse than CIL-R in `Town03` and some other new environments such as `Town05-Dense`. Generally, INT and CIL-R have similar low-level performances in outdoor driving areas, especially in heavy traffic. This is because they only use visual perception, which often has troubles in tough environments such as *StormDark*. By contrast, PMP-net achieves a much higher SR in all evaluation setups, which indicates a superior generalization capability. In particular, the SR increases to 100% in all environments for the empty traffic, and to 72∼88% for regular and dense traffic.

Quality: We use WL and OVSP to evaluate the driving quality of different models. Due to the lack of concrete direction guidance, CIL and CIL-R both have high WL values (3.35∼66.05%). Specifically, they often navigate the vehicle to drive in the correct direction but in the wrong lanes. With the help of the global route information, the models are able to drive more accurately, as we can see by the WL values for INT and PMP, which are all close to 0%. However, INT tends to control the vehicle to drive at high speeds without slowing down at intersections. This unsafe phenomenon leads to high values of OVSP for INT

(5.85∼37.12%). While PMP still performs well on this metric (0.0∼0.4%).

Generally, the remarkable improvements of PMP-net on the benchmark w.r.t. the other three baselines confirm that our proposed model can effectively learn and deploy the driving knowledge in complex dynamic environments.

3) Qualitative Analysis: Fig. 5 shows the qualitative results of PMP-net. When there are no obstacles ahead on straight roads, our model drives relatively fast, at about 40 km/h (Fig. 5-(c)). When taking turns or following road agents, our model reasonably slows down as a human driver would, as shown in Fig. 5-(a, b, d, f, i). In addition, we show some results in extreme conditions. In Fig. 5-(e), the traffic is heavy with many vehicles driving at an intersection. Although the model is directed to turn right, it applies full brake as another vehicle blocks the road ahead. Moreover, in Fig. 5-(g,h), we set dense traffic on a dark night where slow-moving obstacles are ahead of the ego-vehicle. In these scenes with limited vision, PMP-net is also able to drive safely by reducing the throttle to slow down when changing/following lanes. Furthermore, the most challenging scene is shown in Fig. 5-(m). In the *StormDark* environment, there is a small child running across the road abruptly without any previous notice. For this scene, it is difficult to raise alarm even for a human driver because the surroundings cannot be seen clearly. Surprisingly, our model slows down timely by applying brake to avoid an accident. Fig. 5-(k) is another similar scenario. For interpretation, the planned motion distribution of Fig. 5-(m) is attached, where we can see that the planned speed drops rapidly within a short horizon (∼0.5 s) with low variance. We accredit such prominent performance to our multimodal and probabilistic setup. More related driving behaviors are shown in supplementary videos.[3]

V. Conclusion

In this letter, to realize autonomous driving in outdoor dynamic environments, we proposed a deep navigation model named PMP-net, which is based on multimodal sensors (a camera, lidar and radar) and probabilistic end-to-end control. We collected a large-scale driving dataset in the CARLA simulator and trained the model with imitation learning. In order to fully evaluate the driving performance, we further proposed a new online benchmark *DeepTest*, of which the environmental complexity has not been previously considered. By setting varied illumination, weather and traffic conditions in different towns, we showed that our model achieves excellent driving and generalization performance in both unseen urban and rural areas with extreme weather and heavy traffic with dynamic objects (e.g., vehicles, bicyclists and jaywalkers).

To further extend PMP-net for real autonomous vehicles, the *reality gap* should be considered. 1) For discrepancy of sensory input, we can finetune the model with real-world data. The sensor readings of lidar and radar are more consistent than those of a camera with real/simulated deployments, which can help regularize the finetuning process for *domain adaption*. 2) For discrepancy of driving platforms, we can adjust the parameters of the PID controller to adapt to different vehicle properties [14], due to the *modular* design of our network.

[3] Demo videos and a supplementary file including model parameters and benchmark visualization are available at https://sites.google.com/view/pmpnet/

References

[1] J. Leonard *et al.*, “A perception-driven autonomous urban vehicle,” *J. Field Robot.*, vol. 25, no. 10, pp. 727–774, 2008.

[2] E. D. Dickmanns, “The development of machine vision for road vehicles in the last decade,” in *Proc. IEEE Intell. Veh. Symp.*, 2002, vol. 1, pp. 268–281.

[3] M. Bojarski *et al.*, “End to end learning for self-driving cars,” 2016, *arXiv:1604.07316*.

[4] W. Gao, D. Hsu, W. S. Lee, S. Shen, and K. Subramanian, “Intention-net: Integrating planning and deep learning for goal-directed autonomous navigation,” in *Proc. Conf. Robot Learn.*, 2017, pp. 185–194.

[5] F. Codevilla, M. Miiller, A. López, V. Koltun, and A. Dosovitskiy, “End-to-end driving via conditional imitation learning,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 1–9.

[6] H. Xu, Y. Gao, F. Yu, and T. Darrell, “End-to-end learning of driving models from large-scale video datasets,” in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 2174–2182.

[7] F. Codevilla, E. Santana, A. M. Lopez, and A. Gaidon, “Exploring the limitations of behavior cloning for autonomous driving,” in *Proc. IEEE Int. Conf. Comput. Vision*, Oct. 2019, pp. 9328–9337.

[8] L. Tai, P. Yun, Y. Chen, C. Liu, H. Ye, and M. Liu, “Visual-based autonomous driving deployment from a stochastic and uncertainty-aware perspective,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 2622–2628.

[9] A. Amini, G. Rosman, S. Karaman, and D. Rus, “Variational end-to-end navigation and localization,” in *Proc. Int. Conf. Robot. Autom.*, May 2019, pp. 8958–8964.

[10] S. Hecker, D. Dai, and L. Van Gool, “End-to-end learning of driving models with surround-view cameras and route planners,” in *Proc. Eur. Conf. Comput. Vision*, 2018, pp. 435–453.

[11] P. Karkus, X. Ma, D. Hsu, L. Kaelbling, W. S. Lee, and T. Lozano-Perez, “Differentiable algorithm networks for composable robot learning,” in *Proc. Robot.: Sci. Syst.*, Jun. 2019.

[12] M. Mueller, A. Dosovitskiy, B. Ghanem, and V. Koltun, “Driving policy transfer via modularity and abstraction,” in *Proc. 2nd Conf. Robot Learn.*, Oct. 2018, vol. 87, pp. 1–15.

[13] P. Cai, Y. Sun, Y. Chen, and M. Liu, “Vision-based trajectory planning via imitation learning for autonomous vehicles,” in *Proc. IEEE Intell. Transp. Syst. Conf.*, Oct. 2019, pp. 2736–2742.

[14] P. Cai, Y. Sun, H. Wang, and M. Liu, “VTGNet: A vision-based trajectory generation network for autonomous vehicles in urban environments,” 2020, *arXiv:2004.12591*.

[15] M. Bansal, A. Krizhevsky, and A. Ogale, “Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst,” in *Proc. Robot.: Sci. Syst.*, Jun. 2019.

[16] A. Pokle *et al.*, “Deep local trajectory replanning and control for robot navigation,” in *Proc. Int. Conf. Robot. Autom.*, May 2019, pp. 5815–5822.

[17] X. Huang, S. G. McGill, B. C. Williams, L. Fletcher, and G. Rosman, “Uncertainty-aware driver trajectory prediction at urban intersections,” in *Proc. Int. Conf. Robot. Autom.*, 2019, pp. 9718–9724.

[18] F. Codevilla, A. M. Lopez, V. Koltun, and A. Dosovitskiy, “On offline evaluation of vision-based driving models,” in *Proc. Eur. Conf. Comput. Vision*, Sep. 2018, pp. 236–251.

[19] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, “From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots,” in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2017, pp. 1527–1533.

[20] P. Cai, X. Mei, L. Tai, Y. Sun, and M. Liu, “High-speed autonomous drifting with deep reinforcement learning,” *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1247–1254, Apr. 2020.

[21] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Jun. 2016, pp. 770–778.

[22] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, “Probabilistic trajectory prediction with gaussian mixture models,” in *Proc. IEEE Intell. Veh. Symp.*, 2012, pp. 141–146.

[23] D. A. Pomerleau, “ALVINN: An autonomous land vehicle in a neural network,” in *Proc. Advances Neural Inf. Process. Syst.*, 1989, pp. 305–313.

[24] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *Proc. 1st Annu. Conf. Robot Learn.*, Nov. 2017, vol. 78, pp. 1–16.