FlowDriveNet: An End-to-End Network for Learning Driving Policies from Image Optical Flow and LiDAR Point Flow

Shuai Wang[1], Jiahu Qin[1], Menglin Li[1] and Yaonan Wang[2]

Abstract— **Learning driving policies using an end-to-end network has been proved a promising solution for autonomous driving. Due to the lack of a benchmark driver behavior dataset that contains both the visual and the LiDAR data, existing works solely focus on learning driving from visual sensors. Besides, most works are limited to predict steering angle yet neglect the more challenging vehicle speed control problem. In this paper, we propose a novel end-to-end network, FlowDriveNet, which takes advantages of sequential visual data and LiDAR data jointly to predict steering angle and vehicle speed. The main challenges of this problem are how to efficiently extract driving-related information from images and point clouds, and how to fuse them effectively. To tackle these challenges, we propose a concept of point flow and declare that image optical flow and LiDAR point flow are significant motion cues for driving policy learning. Specifically, we first create an enhanced dataset that consists of images, point clouds and corresponding human driver behaviors. Then, in FlowDriveNet, a deep but efficient visual feature extraction module and a point feature extraction module are utilized to extract spatial features from optical flow and point flow, respectively. Additionally, a novel temporal fusion and prediction module is designed to fuse temporal information from the extracted spatial feature sequences and predict vehicle driving commands. Finally, a series of ablation experiments verify the importance of optical flow and point flow and comparison experiments show that our flow-based method outperforms the existing image-based approaches on the task of driving policy learning.**

Supplementary Material

The supplementary video is provided at `https://youtu.be/cPjQcE_fDi4` and the project's dataset, code and trained models will be made available at `https://github.com/wsustcid/FlowDriveNet`.

I. Introduction

Traditional autonomous driving system usually consists of a series of complicated modules, such as localization, perception, planning, control, etc [1], [2]. However, the tight internal dependencies between these modules increase the risk of the system, and a small error from a single module may trigger a chain reaction and eventually cause the failure of the entire task [3]. In order to reduce the modules' internal dependencies, some end-to-end learning-based methods have been proposed, which aims to learn driving policies directly from sensory inputs and has achieved impressive results in recent years [4].

[1]S. Wang, J. Qin and M. Li are with the Department of Automation, University of Science and Technology of China, Hefei 230027, China. `jhqin@ustc.edu.cn`

[2]Y. Wang is with the College of Electrical and Information Engineering, Hunan University, Changsha 410082, China.

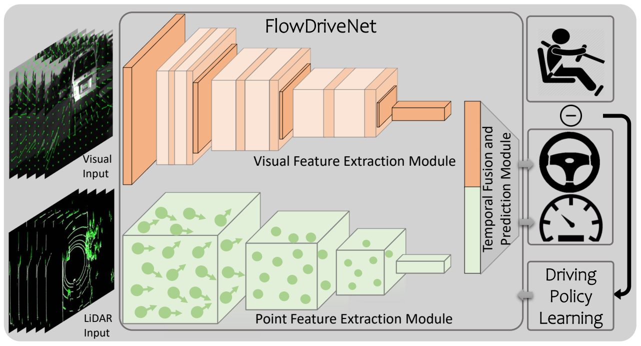
FlowDriveNet
Visual Input
Visual Feature Extraction Module
LiDAR Input
Point Feature Extraction Module
Temporal Fusion and Prediction Module
Driving Policy Learning

Fig. 1. An illustration of our driving policy learning system. FlowDriveNet takes image sequences combined with optical flow and point cloud sequences combined with point flow as inputs. The driving policies are learned from human driver behaviors to predict the vehicle driving commands.

Most end-to-end driving systems [5], [6] have shown satisfactory results in the task of predicting steering angle from visual sensors. However, current driving models have only ten or so layers while the state-of-the-art (SOTA) visual perception models [7], [8], [9] in the field of computer vision have been developed up to hundreds of layers, which hinders the further improvements of the model performance. In addition, in order to make the vehicle acquire the full self-driving capability, the driving model has to predict not only the future steering angle but also the speed control command. The later is much more difficult and has not been well solved in the existing vision-based end-to-end learning systems.

Intuitively, speed control requires the driving model to estimate the relative motion of the vehicle and its surrounding objects from current or historical data. Thus, it is a challenging task for the current models that infer speeds solely from visual images [10], [11]. Different from visual sensors, Light Detection And Ranging (LiDAR) sensor directly provides precise geometrical information of the 3D environments by collecting point cloud data. Thus predicting speeds from point clouds will be a much simpler matter. However, point cloud data has not been well used in the current end-to-end driving systems, partly because of the lack of LiDAR datasets which can be used for driving policy learning.

In this paper, we propose a novel end-to-end driving policy learning network named FlowDriveNet, which aims to simultaneously predict steering angle and vehicle speed from both image sequences and point cloud sequences. First, image optical flow and LiDAR point flow are computed from the consecutive input frames to provide the model with more explicit motion cues. Second, a deep but efficient

Manuscript 716 submitted to 2021 IEEE International Conference on Robotics and Automation (ICRA). Received October 30, 2020.

Visual Feature Extraction (VFE) module and a Point Feature Extraction (PFE) module are designed to extract driving-related features from the multimodal input data. Then, a Temporal Fusion and Prediction (TFP) module is added to the end of the network to explore the temporal correlations from the extracted features and output the vehicle driving control commands. Finally, the contributions of each module are verified by the ablation study. The contributions of this paper are summarized as follows:

- We create a benchmark dataset that contains time-synchronized image sequences, point cloud sequences and corresponding human driver behaviors to facilitate the research of the LiDAR-based driving policy learning.
- To the best of our knowledge, this is the first work that combines optical flow and point flow as inputs to learn driving policies in an end-to-end manner, which simultaneously solves the task of steering prediction and speed prediction in one network.
- The rich ablation experiments verify the importance of optical flow and point flow. In addition, comparison experiments and the model visualization results demonstrate the superiority and reliability of our model.

The remainder of this paper is organized as follows. Section II reviews the development of related works in the past years and Section III states the dataset used in this paper. Section IV presents the architectures of our proposed model. Ablation study and comparison experiments are shown in Section V and this work is concluded in Section VI.

II. Related Work

End-to-end driving system adopts the idea of behavioral cloning that aims to construct a direct mapping from sensory inputs to driving actions by learning from human driver behaviors. Early works mainly focus on predicting steering angle from instantaneous visual images, and the first work can be traced back to the work of ALVINN (Autonomous Land Vehicle In a Neural Network) [12] in the late 1980s. ALVINN followed the connectionist approach and built a 3-layer neural network for the task of road following.

With the development of deep learning, LeCun et al. successfully transfer their LeNet [13] which originally designed for handwritten digit recognition to an end-to-end obstacle avoidance system [14]. In 2016, the end-to-end driving system began its golden age with the impressive results achieved by the *NVIDIA*'s PilotNet [5], which learns to drive in highways and even unpaved roads by predicting steering commands solely from a single front-facing camera. Following this idea, [15] controlled a quadrotor following forest trails by training a similar Convolutional Neural Network (CNN). Then in 2018, [16] tried to predict steering angles from the synchronized event frames captured by the event camera and also provided their datasets in [17], [18].

More recently, a number of researches have recognized the importance of Spatio-Temporal (ST) data. [6] took the ST data as inputs by computing optical flow from video clips and achieved the SOTA results on the task of steering prediction. Similarly, [19] encoded the ST information in the image sequences by using the Convolutional Long Short-Term Memory (Conv-LSTM) layers, and the future driving actions were also used to guide the network training.

However, all the aforementioned works only achieve steering prediction, which is not sufficient for the vehicle control. Speed commands are indispensable in real traffic scenes. In order to predict speed commands of the car, the driving model needs to capture the motion information of the scene. Thus, [10] and [11] utilized ST-Conv, Conv-LSTM and LSTM to extract spatial and temporal visual cues from compiled video frames, and [20] predicted steering angle and vehicle speed from surround-view cameras to get more complete description of the dynamic environments. Obviously, inferring the motion of the objects from visual sensors is not a direct and efficient way.

Instead of predicting the desired steering angle or vehicle speed that can be directly utilized to the vehicle control, some works also tried to predict the intermediate affordance information [21], a distribution of the future feasible actions (such as straight, stop, left turn and right turn) [22], or a collision probability of the vehicle [23] to build a vehicle controller. However, LiDAR sensor has not been used in all the above-mentioned works and the problem of the joint prediction of steering angle and speed has not been perfectly solved.

III. Dataset

A. Udacity Dataset

Udacity CH2[1] is a popular vision-based driver behavior dataset. Though LiDAR data is also contained in this dataset, it has not been utilized in current end-to-end driving systems, since there is no research or relevant tools that try to extract LiDAR data from the ROS (Robot Operating System) bag files provided in *CH2*. Besides, the raw point cloud in *CH2* cannot be directly used for driving policy learning, and the main difficulty lies in the following three aspects:

Distortion The accuracy of the point position in the point cloud is critical to driving policy learning. However, the point cloud in *CH2* is distorted caused by the bump in the road or vehicle jittering, which may interfere with the learning of driving policies.

Noise and Range Diversity The cloud boundary in *CH2* ranging from [-140,-120,-15] to [140,120,30] in $[x,y,z]$ directions, respectively, which bring additional challenges to the model perception since too many isolated and outlier points increase the noise of the LiDAR data.

Density Variation The number of points in the raw point cloud of *CH2* ranging from 7500 to 27000, which hinders the application of PointNet-based method [24], [25] which requires a fixed number of points as input.

B. Udacity CH2-LiDAR

Based on the above discussion, the raw point cloud data in *CH2* need to be preprocessed to meet the requirements

[1]https://www.github.com/udacity/self-driving-car

Manuscript 716 submitted to 2021 IEEE International Conference on Robotics and Automation (ICRA). Received October 30, 2020.

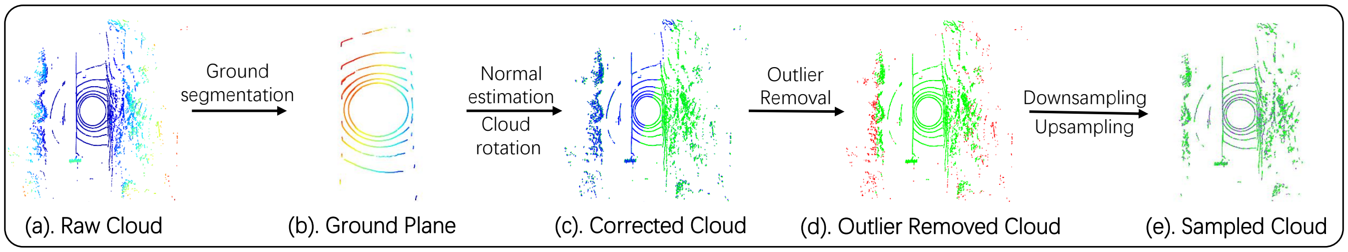

Fig. 2. The point cloud preprocessing pipeline. (a). The raw point cloud is rendered by the height of the point. (b). The points on the ground plane exhibiting different colors mean that the point cloud has distortion. (c). The corrected points are visualized with green color. (d). The red points are the outlier and the isolated points. (e). Finally, the sampled points are shown in green.

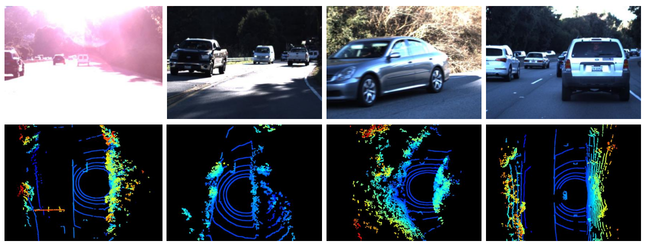

Fig. 3. Samples of driving images (first row) and associated point clouds (second row) in the *CH2-LiDAR* dataset. Apparently, the LiDAR data has a better description of the 3D environments than the visual data.

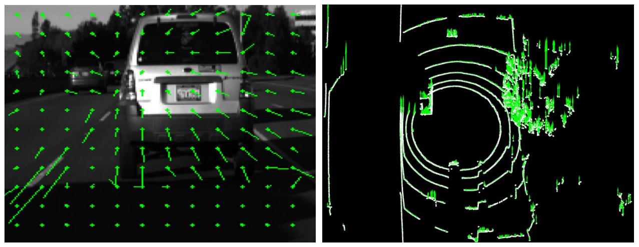

Fig. 4. Visualization of optical flow (left) and point flow (right). The motion of pixels and points is represented by the green arrow.

of driving policy learning. The cloud preprocessing pipeline used in this paper is depicted in Fig. 2.

Cloud Correction For each raw point cloud extracted from *CH2*, we employ a cloud correction procedure to reduce distortions. First, we segment a rough ground plane by selecting the points within a specific height range. Specifically, we select $P_z \in$ [-5,-1.5] where the z axis perpendicular to the ground. Then we adopt the RANSAC (RANdom SAmple Consensus) [26] algorithm to estimate the normal vector of the ground plane, which should align with the Z axis of the LiDAR coordinate. Thus the rotation matrix between the normal vector and Z axis can be computed. Finally, the raw cloud can be corrected by rotating it with the rotation matrix.

Outlier Removal To remove the outlier points, we crop the corrected cloud within the range of [-60,60], [-30,30], and [-5,10] along with the x,y,z axis, respectively. Furthermore, when the standard error of the distances between a point and its 5 nearest neighbors larger than a predefined threshold, the point is deemed as isolated and will be removed.

Cloud Sampling To make all the final point cloud has the same number of points (20000 points in our dataset), we randomly upsampling or downsampling the outlier removed cloud according to the number of points in each point cloud.

Finally, we obtain the processed driving dataset, named *CH2-LiDAR*, which contains curated driving images, corrected point clouds, synchronized control messages, and other sensor information. As shown in Fig. 3, LiDAR data has a better description of the 3D environments than visual images. In Addition, the comparison of *CH2-LiDAR* with the existing public driving datasets is demonstrated in Table I.

TABLE I
COMPARISON OF EXISTING PUBIC DRIVING DATASETS.

Datasets	Image	LiDAR	GPS & IMU	Behaviors
Drive360	✓	✗	✓	✓
Comma.ai	✓	✗	✓	✓
BDDV	✓	✗	✓	✗
Cityscapes	✓	✗	✓	✗
KITTI	✓	✓	✓	✗
CH2-LiDAR	✓	✓	✓	✓

IV. METHODOLOGY

FlowDriveNet can be conceptually separated into three complementary modules, a VFE module that benefits from the SOTA visual perception model [8], a PFE module that built upon the popular point cloud learning network [24], [25], and a well-designed TFP module to fuse the ST features extracted from above two modules.

A. *Input preprocessing*

The sequential inputs should be processed to get the optical flow and point flow before being fed into the VFE module and the PFE module, respectively.

Image sequence with optical flow We first load an image sequence in the gray-scale format to compute the optical flow between two successive images. Gunner Farneback's algorithm [27] is used to get the dense optical flow, which is a 2D vector field for all the pixels of the image. As visualized in Fig. 4, the green arrow shows the movement of the pixel. Then for each image and its corresponding optical flow, the top 100 rows are cropped to remove the useless sky

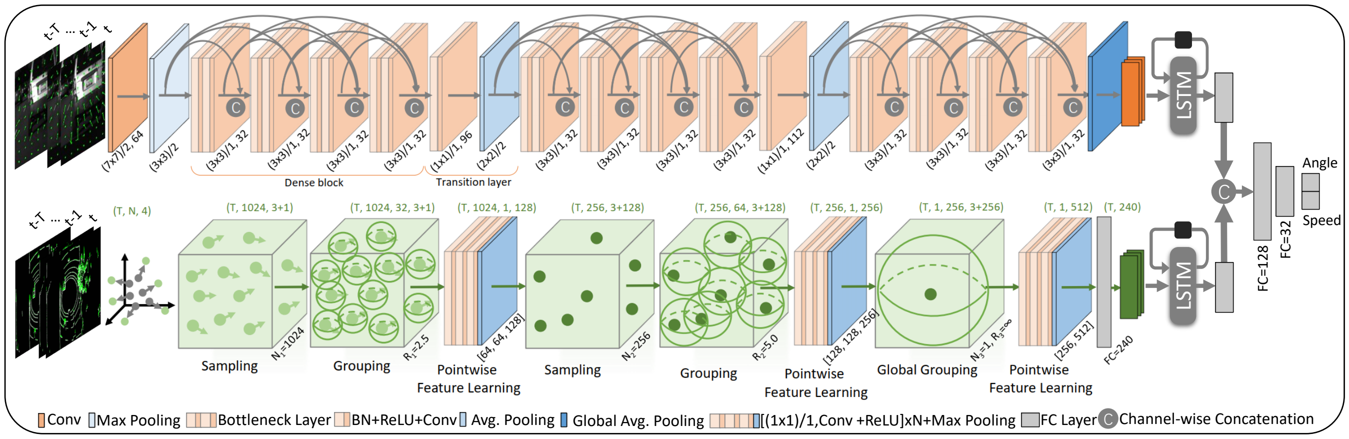

Fig. 5. FlowDriveNet architecture. This model takes both image sequence with optical flow and point cloud sequence with point flow as input. A densely connected convolutional VFE module (upper part of the figure) and a hierarchical PFE module (lower part of the figure) are applied to these two kinds of sequential inputs, respectively. Then, the extracted temporal visual features and point features are respectively processed by an LSTM layer. Finally, three fully connected layers are stacked together to fuse these two kinds of information and predict driving commands.

pixels. Finally, the preprocessed gray-scale image sequences are stacked with their optical flow and resized to the shape of 200×200. The final visual input is a 4D tensor with the shape of $(T, 200, 200, 3)$ where T is the length of the sequence. Input normalization is also performed by subtracting the mean and dividing the standard deviation.

Point cloud sequence with point flow Similar to the optical flow, we also expect to find the explicit motion cues of the surrounding objects from the rich geometrical information of the 3D environment provided by the LiDAR point cloud. Thus, based on the preprocessed point cloud described in Section III-B, for each point of the point cloud perceived at time t, we search the closest point from its previous point cloud at time t-1 and compute their euclidean distance, and the distance is defined as the point flow. As shown in Fig. 4, the movements of the points are explicitly represented by the point flow. Finally, the preprocessed point cloud sequence with the length of T is stacked with their point flow and form a 3D tensor with the shape of $(T, 20000, 4)$.

B. Visual Feature Extraction Module

As depicted in Fig. 5, the multimodal visual input (image sequences and their optical flow) are firstly processed by a 7×7 Conv layer with stride 2 to detect some driving-related low-level features and a 3×3 max pooling layer with stride 2 to reduce the spatial size of the feature maps. Then three dense blocks are used to learn more abstract features. Each dense block is composed of 4 densely connected bottleneck layers. The bottleneck layer is designed as BN-ReLU-Conv(1×1)-BN-ReLU-Conv(3×3), in which the batch normalization and the 1×1 Conv layer with 128 output filters are added before the 3×3 Conv layer to improve the network efficiency. Dense connection denotes that each layer in the dense block takes all the preceding feature maps as input, which strengthens feature propagation and encourages feature reuse. Between every two dense blocks, a transition layer is utilized to further reduce the depth and spatial size of feature maps by setting the number of output filters in the 1×1 Conv layer as half number of input feature maps and the pooling stride as 2 in the 2×2 average pooling layer. Finally, a global average pooling layer is used to aggregate the extracted ST features with the shape of $(T, 5, 5, 240)$ in the spatial domain, and generate a temporal feature with the shape of $(T, 240)$ to be further processed by the TFP module.

C. Point Feature Extraction Module

As shown in Fig. 5, the multimodal LiDAR input with the shape of $(T, N, 3+1)$ is created from T frames of preprocessed N unordered points with 3 position coordinates and 1 point flow value. Initially, the PFE module takes this 3D tensor as input, and a sampling layer selects 1024 points by applying the Farthest Point Sampling (FPS) algorithm on each frame. Intuitively, each selected point defines the centroid of a local region. Then for each frame, the grouping layer constructs 1024 overlapped neighborhood balls with the radius $R_1 = 2.5$ and sampled 32 local points in each ball. These local points combined with their point flow values are further processed by a pointwise feature learner. Specifically, the pointwise feature learner uses a set of weight shared point-wise perceptrons to learn a spatial encoding of each local region and applies the max pooling operation along the local point axis to make the learned local feature invariant to point permutation. The above process is repeated twice by setting the sampling number $N_2 = 521, N_3 = 1$, and the grouping radius $R_2 = 5.0, R_3 = \infty$ to extract higher-level features in a hierarchical way. Finally, a fully connected layer with a size of 240 is utilized to generate a temporal feature with the shape of $(T, 240)$ to be further used in the TFP module.

To encourage feature reuse, the relative coordinates of points in each local region are used by applying the coordinate transform, in which the centroid point is taken as the origin of the local coordinate system.

TABLE II
PERFORMANCE OF DIFFERENT COMBINATIONS OF PERCEPTION MODULES WITH VARIOUS INPUT VARIATIONS.

Module	Input Variation					Evaluation Metrics			
	Gray Image	Optical Flow	Point Cloud	Point Flow	Sequence	Angle	Speed	Average	R^2
Constant baseline	–	–	–	–	–	0.1065	0.1858	0.1462	-0.384
VFE	✓	–	–	–	–	**0.0693**	0.1544	0.1119	0.178
VFE	✓	✓	–	–	–	**0.0431**	0.1431	0.0931	0.394
VFE	✓	✓	–	–	✓	**0.0367**	0.1568	0.0968	0.312
PFE	–	–	✓	–	–	0.0745	**0.1914**	0.1330	-0.183
PFE	–	–	✓	✓	–	0.0558	**0.0914**	0.0749	0.638
PFE	–	–	✓	✓	✓	0.0546	**0.0554**	0.0550	0.792
VFE+ PFE + TFP	✓	✓	✓	✓	✓	**0.0295**	**0.0507**	**0.0401**	**0.896**

TABLE III
COMPARISON RESULTS OF FLOWDRIVENET WITH EXISTING DRIVING MODELS. RGB, GRAY, XYZ, F, -T REPRESENT COLOR IMAGE, GRAY-SCALE IMAGE, POINT CLOUD, FLOW DATA AND SEQUENTIAL DATA, RESPECTIVELY. AUG MEANS DATA AUGMENTATION IS UTILIZED.

Module	Input Modal	Evaluation on Val. Set		Evaluation on Test Set				Num.	Num.	FPS
		Angle	Speed	Angle	Speed	Average	R^2	Layers	Params.	
PilotNet [5]	RGB	0.0587	–	0.0575	–	–	0.708	10	1.6×10^6	282
MSTNet [6]	GRAYF-T	**0.0033**	–	0.0550	–	–	0.734	9	2.4×10^6	279
PointNet [24]	XYZ	–	0.0555	–	0.1702	–	-0.483	9	8.1×10^5	106
DroNet [23]	GRAY	0.0037	**0.0115**	0.1171	0.1747	0.1459	-0.386	9	$\mathbf{3.2 \times 10^5}$	**369**
FlowDriveNet (Aug)	GRAYF-XYZF-T	0.0165	0.0262	**0.0295**	**0.0445**	**0.0370**	**0.912**	35	1.5×10^6	22.8

D. Temporal Fusion and Prediction Module

In the TFP module, the extracted temporal visual feature and point feature are separately processed by an LSTM layer to capture the motion of the vehicle and its surrounding objects. Then the outputs of the two LSTM layers are concatenated together and are further fused by two fully connected (FC) layers. Finally, an FC layer with 2 output units predicts the steering angle and speed as the vehicle control commands.

It is worth mentioning that we also tried to fuse the visual and the point feature first, and then use an LSTM layer to extract motion cues from the fused feature, which performs worse than the current scheme.

V. EXPERIMENTS

In this section, we evaluate FlowDriveNet with extensive experiments on the *CH2-LiDAR* dataset.

A. Ablation Study

Model Configurations To analyze the improvements gained by each type of motion cues combined with our refinement modules, we first build a constant baseline that always outputs 0 as the model prediction value. Then we train the VFE module with three types of input variations, i.e., the single-frame gray image, the single-frame gray image stacked with its optical flow, and the gray image sequence combined with optical flow. Besides, we utilize an FC layer with two output units to predict driving commands after the VFE module and an LSTM layer is included when the module adopts sequential inputs. For the PFE module, we apply a similar configuration and take the single-frame 3D point cloud, the single-frame 3D point cloud stacked with its point flow, and the point cloud sequence combined with point flow as the module inputs, respectively. Finally, FlowDriveNet that combines the TFP module with the VFE module and PFE module is trained by taking image sequences and point cloud sequences stacked with their flow data as inputs.

Training Configuration All the models are trained without data augmentation and evaluated on the test set with the metric of Root Mean Square Error (RMSE) and R-Squared (R^2). The Adam optimizer [28] with the parameter of $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$ is utilized to minimize the loss of Mean Squared Error (MSE) and the initial learning rate is set to 10^{-4}. The length of input sequences used in our experiments is 5. To prevent overfitting, the training is stopped when the evaluation metric on the validation set has stopped improving for 20 consecutive epochs.

Results From the results presented in Table II, we can draw three main conclusions. (a). The explicit motion cues provided by the optical flow and point flow are respectively beneficial for the prediction of steering angle and vehicle speed. (b). The temporal information provided by the sequential inputs can further improve the module performance.

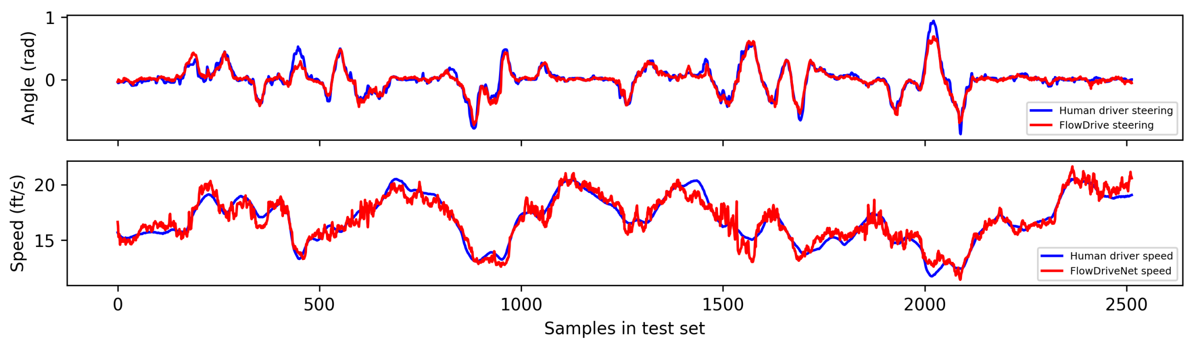

Fig. 6. The prediction results of FlowDriveNet compared with the recorded human driving behaviors. Apparently, all the driving behaviors have been learned successfully by our driving model, but the speed prediction needs to be further improved to make vehicle speed control as smooth as the steering control.

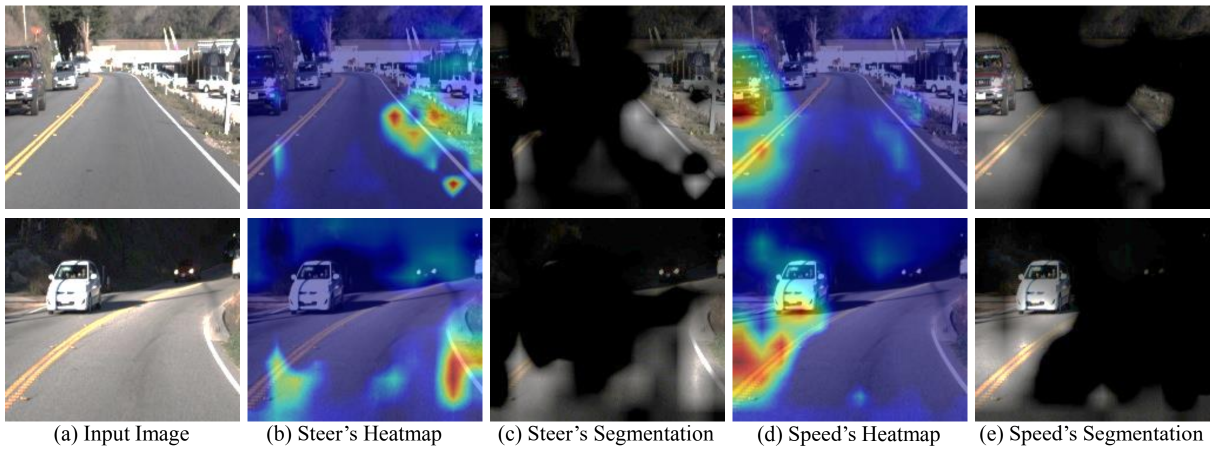

Fig. 7. The model visualization results on the test data. We use the activation heatmap and segmented image (inactive pixels are painted in black) to visualize the driving model's attention. As shown in (b) and (c), on the task of steering prediction, the lane lines activate our driving model more than the rest of the image. (d) and (e) show that both the surrounding vehicles and lane lines contribute to the speed prediction.

(c). With the help of three complementary visual perception module, LiDAR perception module, and temporal fusion module, our FlowDriveNet successfully combines the benefits of optical flow, point flow, and sequential inputs.

B. Comparison Experiments

To further verify the advantages of FlowDriveNet, we compare our model with the SOTA vision-based driving models [5], [6], [23] and the popular point cloud learning network [24]. As shown in Table III, FlowDriveNet outperforms the other driving models on the evaluation metrics of the test data while other models have a better performance on the validation data, which also demonstrates that our model is indeed learning driving policies rather than fitting data. Although FlowDriveNet performs poorly on the inference time, it can also ensure the driving system running in real-time since the sampling frequency of the LiDAR sensor cannot be higher than 10HZ. The prediction results of FlowDriveNet on the test data are displayed in Fig 6, which shows that the human driving policies have been learned successfully by FlowDriveNet but the speed control needs to be further improved to make it as smooth as the steering control.

C. Qualitative Results

To better understand the driving policies learned by our FlowDriveNet, we employ the model visualization technique presented in [29], [30]. As shown in Fig. 7, the lane lines are the most important driving information, which contribute to both the steering and the speed prediction. In addition, the model also concentrates on the surrounding vehicles when making decisions on the speed control, which is consistent with human driving behaviors.

VI. Conclusion

This paper investigates the benefits of learning driving policies from optical flow and point flow. A benchmark driving behavior dataset named *CH2-LiDAR* is firstly created based on the *Udacity-CH2* dataset to promote the research of learning driving from LiDAR data. Besides, our FlowDriveNet solves the challenging problem of jointly predicting steering angle and vehicle speed in one network and verifies the importance of spatial and temporal motion cues extracted from the multimodal sequential inputs. In the future, we will deploy our driving model to a real self-driving car and further improve the model performance on the speed prediction.

Manuscript 716 submitted to 2021 IEEE International Conference on Robotics and Automation (ICRA). Received October 30, 2020.

References

[1] Levinson, Jesse, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J. Zico Kolter et al, "Towards fully autonomous driving: Systems and algorithms," *In 2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 163-168.

[2] Yurtsever E, Lambert J, Carballo A, et al., "A survey of autonomous driving: Common practices and emerging technologies", *IEEE Access*, vol. 8, pp. 58443-58469, 2020.

[3] Siheng Chen, Baoan Liu, Chen Feng, Carlos Vallespi-Gonzalez and Carl Wellington, "3d point cloud processing and learning for autonomous driving", *IEEE Signal Processing Magazine*, Special Issue on Autonomous Driving, pp. 1-24, 2020.

[4] S. Grigorescu, B. Trasnea and T. Cocias, "A survey of deep learning techniques for autonomous driving", *Journal of Field Robotics*, vol. 37, pp. 362-386, 2020.

[5] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao and K. Zieba, "End to end learning for self-driving cars", *CoRR*, vol. abs/1604.07316, 2016.

[6] M. Abou-Hussein, S. H. Mller and J. Boedecker, "Multimodal Spatio-Temporal Information in End-to-End Networks for Automotive Steering Prediction," *in IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8641-8647.

[7] He, K., Zhang, X., Ren, S. and Sun, J, "Deep residual learning for image recognition," *in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770-778.

[8] G. Huang, Z. Liu, K. Q. Weinberger, and L. Maaten, "Densely connected convolutional networks," *in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4700-4708.

[9] Zoph, B., Vasudevan, V., Shlens, J. and Le, Q. V, "Learning transferable architectures for scalable image recognition," *In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2018, pp. 8697-8710.

[10] Chi, Lu and Yadong Mu, "Deep Steering: Learning End-to-End Driving Model from Spatial and Temporal Visual Cues", *CoRR*, vol. abs/1708.03798, 2017.

[11] Z. Yang, Y. Zhang, J. Yu, J. Cai and J. Luo, "End-to-end multi-modal multi-task vehicle control for self-driving cars with visual perceptions," *in 24th International Conference on Pattern Recognition (ICPR)*, 2018, pp. 2289-2294.

[12] D. A. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network", *in Advances in Neural Information Processing Systems*, 1989, pp. 305-313.

[13] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, vol. 86, pp. 2278-2324, 1998.

[14] Y. LeCun, U. Muller, J. Ben, E. Cosatto and B. Flepp, "Off-Road Obstacle Avoidance through End-to-End Learning", *in Advances in Neural Information Processing Systems*, 2006, pp. 739-746.

[15] Alessandro Giusti, Jerome Guzzi, et al., "A machine learning approach to visual perception of forest trails for mobile robots", *IEEE Robotics and Automation Letters*, vol. 1, pp. 661-667, 2016.

[16] A. I. Maqueda, A. Loquercio, G. Gallego, N. Garca and D. Scaramuzza, "Event-based vision meets deep learning on steering prediction for self-driving cars," *in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 5419-5427.

[17] Binas, J., Neil, D., Liu, S. C. and Delbruck, T, "DDD17: End-to-end DAVIS driving dataset", *arXiv preprint*, arXiv:1711.01458, 2017.

[18] Hu, Y., Binas, J., Neil, D., Liu, S. C. and Delbruck, T, "DDD20 End-to-End Event Camera Driving Dataset: Fusing Frames and Events with Deep Learning for Improved Steering Prediction", *arXiv preprint*, arXiv:2005.08605, 2020.

[19] Wu, T., Luo, A., Huang, R., Cheng, H. and Zhao, Y, "End-to-End Driving Model for Steering Control of Autonomous Vehicles with Future Spatiotemporal Features," *In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 950-955.

[20] S. Hecker, D. Dai and L. Van Gool, "End-to-end learning of driving models with surround-view cameras and route planners," *in Proc. of the European Conference on Computer Vision (ECCV)*, 2018, pp. 435-453.

[21] Chen, C., Seff, A., Kornhauser, A., and Xiao, J, "Deepdriving: Learning affordance for direct perception in autonomous driving," *In Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2722-2730.

[22] H. Xu, Y. Gao, F. Yu and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," *in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3530-3538.

[23] A. Loquercio, A. I. Maqueda, C. R. del Blanco and D. Scaramuzza, "DroNet: Learning to fly by driving", *IEEE Robotics and Automation Letters*, vol. 3, pp. 1088-1095, 2018.

[24] C. R. Qi, H. Su, K. Mo and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 652-660.

[25] C. R. Qi, L. Yi, H. Su and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *in Advances in Neural Information Processing Systems*, 2017, pp. 5099-5108.

[26] Fischler, Martin A. and Robert C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography", *Communications of the ACM*, vol. 24, pp. 381-395, 1981.

[27] G. Farneback, "Two-frame motion estimation based on polynomial expansion," *in Scandinavian conference on Image analysis, Springer*, 2003, pp. 363-370.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *In International Conference on Learning Representations (ICLR)*, 2015.

[29] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," *in 2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 618-626.

[30] Chattopadhay, Aditya, Anirban Sarkar, Prantik Howlader, and Vineeth N. Balasubramanian, "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks," *in 2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 839-847.

Manuscript 716 submitted to 2021 IEEE International Conference on Robotics and Automation (ICRA). Received October 30, 2020.