Visual Explanation by Attention Branch Network for End-to-end Learning-based Self-driving

Keisuke Mori[1,*], Hiroshi Fukui[1,*], Takuya Murase[1]
Tsubasa Hirakawa[1], Takayoshi Yamashita[1], Hironobu Fujiyoshi[1]

***Abstract*— Self-driving decides an appropriate control considering the surrounding environment. To this end, self-driving control methods by using a convolutional neural network (CNN) have been studied, which directly input the vehicle-mounted camera image to a network and output a steering directory. However, if we need to control not only steering but also throttle, it is necessary to grasp the state of the car itself in addition to the surrounding environment. Moreover, in order to use CNNs for critical applications such as self-driving, it is important to analyze where the network focuses on the image and to understand the decision making. In this work, we propose a method to solve these problems. First, to control both steering and throttle simultaneously, we propose using the current vehicle speed as the state of the car itself. Second, we introduce an attention branch network (ABN) architecture to a self-driving model, which enables visually analyzing the reason of the self-driving decision making by using an attention map. Experimental results with a driving simulator demonstrate that our method controls a car stably, and we can analyze the decision making by using the attention map.**

I. INTRODUCTION

To develop self-driving cars, it is necessary to understand the surrounding environment. Most studies on developing self-driving cars measure the surrounding environment with a camera or by light detection and ranging (LIDAR) and decide control values [1], [2]. Apart from these approaches, convolutional neural network (CNN)-based vehicle control approaches, which directly output control values with only a camera image as an input, have been investigated [3], [4], [5], [6]. In this approach, we collect in-vehicle camera images and the corresponding control values when human drivers control a vehicle. By training a network with the collected data in an end-to-end manner, a vehicle can be automatically controlled in the same way a human driver would control it. Because the system autonomously trains driving skills, the end-to-end learning approaches have an advantage in which the system structure becomes simple.

For practical self-driving, both steering and throttle should be stably controlled. The steering could be controlled properly by understanding scenes of front-facing camera images. Unlike a steering, the control of a throttle would be difficult when using only the surrounding environmental information. The reason is that video frames captured by an in-vehicle camera are input to a network one by one, which does not adequately represent the state of the vehicle itself, i.e., a vehicle velocity. To consider temporal changes, a method that introduces a long short-term memory (LSTM) has been proposed [4]. This approach only represents the temporal changes of an input video sequence implicitly, and the actual speed changes of a vehicle are not represented explicitly.

Apart from throttle controls, another problem is the difficulty in understanding the reason for network output. For instance, when the network decides right and left turning, acceleration, or deceleration, giving the reason why a car would perform such operations would put passengers at ease. Also, if a traffic accident occurs because of the self-driving system, analyzing the reason of the network output would be of great help in the investigation of the accident.

In this paper, we tackle the above two problems and propose an end-to-end self-driving method. Our method has two characteristics. First, we estimate both steering and throttle simultaneously. To improve throttle control performance, we use the information of the car itself, i.e., the vehicle velocity, in addition to the surrounding environmental information, which can stably control throttle. Second, we generate an *attention map*, which visualizes the region in which the network is focused as a heat map. We build a network on the basis of an attention branch network (ABN) [7]. To apply the ABN structure to a regression problem, we propose a weighted global pooling (WGP) layer. The conventional global average pooling (GAP) averages the entire feature map values, which results in the loss of the important information. In contrast, because the WGP pools the whole feature map by weighted average, it can preserve sufficient information. Experimental results using a driving simulator demonstrate that the proposed method can stably control steering and throttle. Furthermore, we analyze the obtained attention maps.

The contributions of this paper are as follows:

- We estimate both steering and throttle simultaneously. The proposed method uses a velocity as an additional input, which enables stable throttle controls.
- To apply an ABN architecture for a regression problem, we propose the WGP. By introducing the WGP, we can estimate regression values from the attention branch while obtaining attention maps.
- By using the attention map, we can visually explain the reason of the network output in a self-driving control.

II. RELATED WORK

A. Self-driving based on end-to-end learning

Thanks to the recent development of CNNs, self-driving controls with an end-to-end learning approach have been

*Authors are equally contributed.
[1] Chubu University, Kasugai, 487-8501, Japan.

studied [3], [4], [5], [6]. Bojarski *et al.* [3] proposed a CNN architecture that inputs an in-vehicle camera image and outputs a steering angle. To collect the steering angles used for the training, they drive a car for 72 hours. The collected data contain various conditions such as weather, time, and road conditions, which enables the achievement of better control performance under various conditions. Xu *et al.* [4] proposed introducing a long short-term memory (LSTM) to consider time-series variations of an in-vehicle camera movie. Also, Yang *et al.* [6] convert an actual in-vehicle camera image to a virtual image by a generative adversarial network (GAN) to apply a CNN trained with virtual images of a driving simulator to real images. In addition to controlling steering and throttle, our model reveals the reason for the network decisions by using an attention map.

B. Visual explanation

In the field of computer vision, *visual explanation*, analyzing the factor of a network output by generating an *attention map*, has been widely investigated [8], [9], [10], [11], [12]. The visual explanation can be categorized into two types: bottom-up and top-down. Bottom-up approaches generate attention maps from the gradient information. A guided backpropagation [10] and a gradient-weighted class activation mapping (Grad-CAM) [12] have been proposed. These methods obtain the maps by using only the positive values of gradients of a specific class. These are used as a general analysis method of CNNs because these can be applied to any pre-trained network models and an attention map of a specific object class.

Top-down approaches generate attention maps by using response values output from a network. Although this approach requires the building and re-training of a network, we can obtain attention maps for each object class during the forward pass. A class activation mapping (CAM) [9] generates attention maps with response values obtained from a convolutional layer and the connection weights of a fully connected layer. However, CAM tends to decrease classification performance because it needs to replace a fully connected layer with a convolutional layer. To resolve this problem, an attention branch network (ABN) [7] has been proposed. ABN separates a network into a feature extractor and a perception branch and adds an attention branch behind the feature extractor. The feature extractor extracts feature maps from an input image, and the attention branch is based on convolutional layers. The obtained feature maps are used with the attention branch to output an attention map. To output classification results from the attention branch, we put a global average pooling (GAP) layer [13] at the output layer. Then, the feature map from the feature extractor is weighted by the attention map, and we can consider the highly weighted regions to output results.

In this work, we analyze the reason of a network output with regard to end-to-end self-driving by using an ABN framework. However, since the ABN is built for a classification problem, it is impossible to simply apply the ABN to a regression problem that outputs steering and throttle as continuous values. By introducing a WGP, ABN can be applied for a regression problem.

Visual explanation has been introduced in end-to-end self-driving. Bojarski *et al.* [14] visualized the regions in which a CNN focused upon. They added VisualBackProp [15] modules to an original CNN and generated a visualization mask that could be obtained by taking point-wise products to the feature maps of every convolutional layer. The visualization mask highlights regions where convolutions respond. Meanwhile, the proposed method introduced an ABN structure that uses the obtained attention map for the estimation, which can improve control performance.

III. PROPOSED METHOD

To achieve complete self-driving in an end-to-end manner, we propose a network model that estimates both steering and throttle and visually explains the reason of network output. To this end, we introduce the following two ideas. One is to use the vehicle velocity as an additional input. Introducing the velocity enables consideration of both the surrounding environment and the state of a car itself, and it can control steering and throttle accurately. The other is to introduce the ABN framework to obtain an attention map. By introducing the ABN framework, we can estimate the control values while obtaining an attention map for visual explanation. Hereafter, we describe the details of the proposed method.

A. Steering and throttle controls by adding velocity

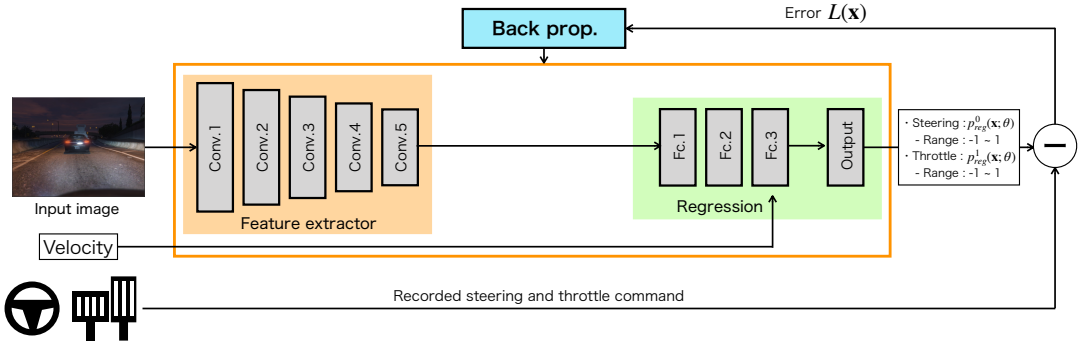
In order to estimate a throttle, we need to train a network while considering the velocity changes of a vehicle. However, conventional end-to-end learning approaches such as [3] cannot consider the velocity changes because they use only an in-vehicle camera image as an input. To resolve this issue, we use a vehicle velocity as an additional input and train the network. By introducing the velocity, we can extract features regarding the internal state of the car itself. The proposed method inputs the velocity at the fully connected layer, as shown in Fig. 1(a). Specifically, the velocity is concatenated with a feature vector obtained from the previous fully connected layer. Then, the network outputs steering and throttle.

Steering and throttle output from the network are normalized to $[-1, 1]$. In case of steering, $[-1, 0)$ indicates turning the steering wheel to the left and $(0, 1]$ to the right. For throttle, $[-1, 0)$ and $(0, 1]$ mean break and acceleration, respectively. The estimated steering and throttle values are obtained by applying the hyperbolic tangent (tanh) function to response values of the output layer $\mathbf{v}^c$. Here, let $p_m^c(\mathbf{x}; \theta)$ be an output value of the network, t_m^c a ground truth, and c a category, where $c = 0$ is for steering and $c = 1$ for throttle. To train a network, we compute a mean squared error $L(\mathbf{x})$ as follows:

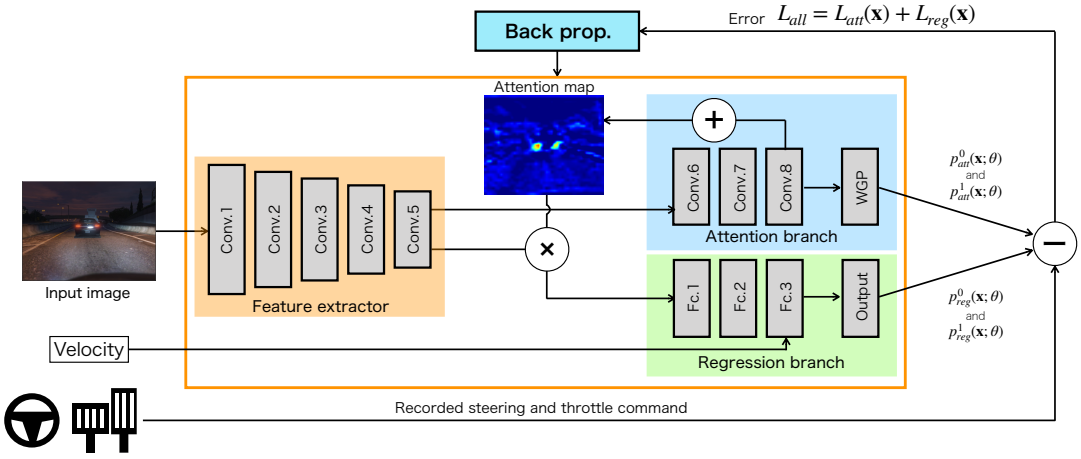
$$p_m^c(\mathbf{x}; \theta) = \tanh(\mathbf{v}^c) \quad (1)$$

$$L(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} \sum_{c=1}^{C} |p_m^c(\mathbf{x}; \theta) - t_m^c|^2, \quad (2)$$

where M is a mini-batch size.



(a) Model of CNN with velocity



(b) Model of our ABN

Fig. 1. Network structures for self-driving system.

B. Attention mechanism for visual explanation

To generate an attention map, we introduce an ABN framework to our self-driving model described in Section III-A. Figure 1(b) shows the details of the proposed ABN structure for self-driving, which consists of a feature extractor, an attention branch, and a regression branch. Note that we refer to the perception branch as a regression branch because we deal with a regression problem. The regression branch has the same structure as the perception branch, and we modified this branch to output regression values instead of classification probabilities. Also, the velocity described in Section III-A is input to a fully connected layer in the regression branch.

A difficulty in applying the ABN for regression problems is inherent in the attention branch. The attention branch needs to be built on the basis of convolutional layers to output an attention map. And, a GAP is used as the output layer of the attention branch. However, the GAP removes information because it takes its average over the entire feature map values. Therefore, it is impractical to use the GAP for the regression problem in the attention branch. In this work, we proposed a weighted global pooling (WGP), which computes the weighted average by using a convolutional layer. As shown in Fig. 2, the WGP applies a convolution whose kernel size is the same as the feature map. Let α be a weight for pooling and $f(\mathbf{x})$ be a feature map for an input $\mathbf{x}$. The

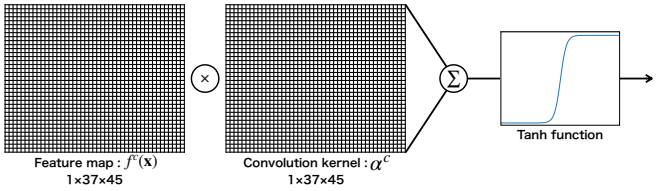


Fig. 2. Overview of the weighted global pooling.

response value of WGP $\mathbf{v}$ is defined by

$$\mathbf{v}^c = \sum_{i=1}^{h}\sum_{j=1}^{w} \alpha^c_{j,i} \cdot f^c_{j,i}(\mathbf{x}), \tag{3}$$

where w and h are the width and height of the feature map, respectively. Because the WGP weights for each element of feature map $f^c_{j,i}(\mathbf{x})$ and outputs a regression value, it is difficult for the WGP to lack information of the pooled feature map.

An attention map is generated by using the feature map obtained from the Conv.8 layer, as shown in Fig. 1(b). Since Conv.8 is applied for the feature map before the WGP is applied, we can obtain two feature maps representing the attentions of steering and throttle. In the proposed ABN, we generate an attention map by adding the two feature maps point-wisely. To train the network, we compute mean squared

TABLE I

AUTONOMY SCORES OF EACH METHOD OVER DIFFERENT DRIVING SCENES.

Method	Throttle					Steering				
	Country road	Urban road				Country road	Urban road			
	Daylight	Daylight	Night	Rainy	All	Daylight	Daylight	Night	Rainy	All
CNN	90.0	76.6	76.6	76.6	79.0	90.0	**100.0**	**100.0**	**93.3**	96.3
CNN+Velocity (Fc.1)	80.0	73.3	70.0	73.3	73.6	80.0	90.0	86.6	86.6	86.3
CNN+Velocity (Fc.2)	90.0	86.6	83.3	83.3	85.4	**95.0**	96.6	96.6	90.0	94.5
CNN+Velocity (Fc.3)	**95.0**	**93.3**	**93.3**	**90.0**	**92.7**	**95.0**	**100.0**	**100.0**	**93.3**	**97.2**
CNN+Velocity (Output)	80.0	83.3	83.3	83.3	82.7	85.0	90.0	90.0	83.3	87.2

(a) Country road (b) Urban road

(c) Night-time (d) Rain

Fig. 3. Examples of driving scenes on the GTAV simulator.

errors for the outputs of attention and regression branches as shown in Eq. (2). Here, we denote the errors of attention and regression branches as L_{att} and L_{reg}, respectively. The loss function is formulated as follows:

$$L_{all}(\mathbf{x}) \quad = \quad L_{att}(\mathbf{x}) + L_{reg}(\mathbf{x}). \tag{4}$$

IV. EXPERIMENTS

In this section, we evaluate the proposed method. First, we evaluate the control performance by adding the vehicle velocity. Then, we analyze the reasons of the network outputs by visualizing attention maps. In addition to the analysis of attention maps, we demonstrate that our method has the potential to explain the reason of the network output linguistically.

Experiments using an actual vehicle in real-world circumstances are costly and risk accidents. Using a simulator environment can resolve the issues. To resolve the issues, we used a driving simulator. Specifically, we built an environment based on Grand Theft Auto V (GTAV), a video game. Figure 3 shows examples of driving scenes on the GTAV simulator. The GTAV simulator can easily configure various settings, e.g., abundant driving environments such as country and urban roads, weather, and time. Moreover, we can drive freely in an extensive world that is about 126 square kilometers. To train networks, we collected in-vehicle camera images and the corresponding values, i.e., steering, acceleration, brake, and speed. The collected images are RGB images whose size are 420 $\times$ 350 pixels. To collect sufficient data, we drive a car in the GTAV simulator for three hours. In the following experiments, we used 30,000 frames for training and 16,556 frames for evaluation.

As a conventional method, we used a CNN architecture proposed in [3]. During the training phase, we used RMSProp+Graves as an optimization method, whose learning rate and exponential decay rate are set to 0.01 and 0.99, respectively. We set the mini-batch size as 4 and train networks in 150 epochs.

A. Evaluations on control performance

Herein, we evaluate the control performance by introducing a vehicle velocity. We used a network shown in Fig. 1(a) as the proposed method to compare the control performance with or without a velocity. As an evaluation metric, we use an autonomy score used in [3], which is defined by

$$\text{autonomy} = \left(1 - \frac{(\sharp \text{ of interventions}) \cdot 6}{\text{elapsed time [sec.]}}\right) \cdot 100. \tag{5}$$

This indicates how many times a human driver intervenes with autonomous controls while a network autonomously controls a vehicle. If the autonomy score is close to 100, the number of the interventions are few, and the network can stably control a vehicle. Moreover, we evaluate the performance in case that we change the layer concatenating a velocity.

Table I shows the autonomy scores of each method over different driving scenes. The scores of the conventional CNN are 79.0% for throttle and 96.3% for steering. Our method introducing a velocity outperforms the conventional CNN. Particularly, when we input a velocity at the Fc.3 layer, the proposed method achieved the highest scores, whose throttle is improved by 13.7%. Because the Fc.1 and Fc.2 layers consist of 1,000 and 100 units respectively, it is thought that a velocity is handled as noise. In contrast, if we input a velocity at the output layer, the efficient feature extraction from the velocity might become difficult. Therefore, it does not contribute to the performance improvement. From these results, we input a velocity at the Fc.3 layer in the following experiments.

B. Visualization results of the network output

Next, we analyze the obtained attention maps. To visualize attention maps, we built the proposed ABN by adding the attention branch structure shown in Tab. II on the conventional CNN [3] as shown in Fig. 1(b). As a comparative

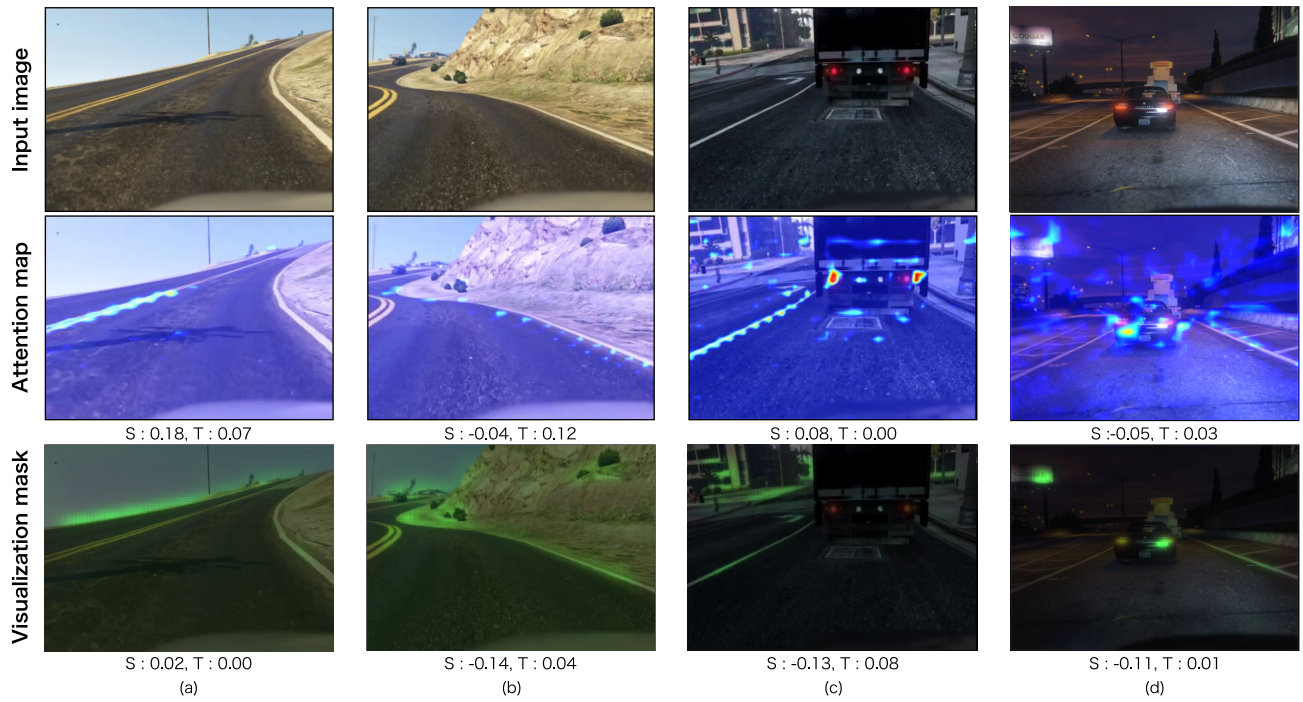


Fig. 4. Examples of the attention map and visualization mask [15]. Values under attention maps and visualization masks are the estimated steering (S) and throttle (T) values in each scene.

TABLE II
DETAILED STRUCTURE OF THE ATTENTION BRANCH

Layer	Detail
1st Conv.	kernel : 1,000 × 1 × 1 activation func. : Leaky ReLU stride : 1
2nd Conv.	kernel : 100 × 1 × 1 activation func. : Leaky ReLU stride : 1
3rd Conv.	kernel : 2 × 1 × 1 activation func. : Leaky ReLU stride : 1
Output (WGP)	kernel : 2 × 37 × 45 activation func. : tanh

visualization method, we used visualization masks [15]. Figure 4 shows examples of the obtained attention maps and visualization masks over different driving scenes. Figure 4(a) shows a scene in which the wheel must be turned to the right. The attention map highlights the center line, and steering is estimated as a positive value, i.e., turn to the right. The visualization mask of the same scene responds to a left side line, although a positive steering value is also estimated. In the scene of Fig. 4(b) in which the wheel must be turned to the left, the estimated steering values are negative, and the right side line is highlighted in both the attention map and visualization masks. These results show that each method focuses on roadway lines to estimate steering, whereas there are differences about highlighted regions.

Figure 4(c) shows a scene in which a car needs to stop. In this scene, the attention map highlights a brake lamp in the front truck. And, the estimated throttle is 0, that is, the network decision makes the car stop. The visualization mask for the same scene does not highlight the brake lamp of the truck ahead and focuses on the building behind the truck. The corresponding throttle is a positive value, which means forward movement. Figure 4(d) is a scene in which a car drives following the front vehicle. Both the attention map and visualization mask focus on the front vehicle. Therefore, to decide throttle, networks pay attention to objects in front of the car. Among them, our method successfully controls a throttle.

C. Caption selection with attention maps

The attention maps highlight regions where a network focuses on estimating control values. This indicates that our method has the potential to explain the reason of the decision making as a language by generating a caption that describes the highlighted region. Therefore, we briefly introduce the results of captions with respect to the reason of the network output. To select captions, we used a fully convolutional localization network (FCLN) [16], which is based on a region proposal network, and it generates captions for each detected region by RPN. In this experiment, we select a caption by selecting a region that is highlighted by the attention map.

Figure 5 shows examples of the selected captions for each scene. In the first row, the throttle changes from 0.05 to -0.15, and the self-driving car tries to stop. In this scene, “man on a red motorcycle” is selected as a caption. This means that the estimated decision is reasoned as “a car on the road.” Likewise, in the second row, the estimated decision is also reasoned as “a car on the road.” From these results, the

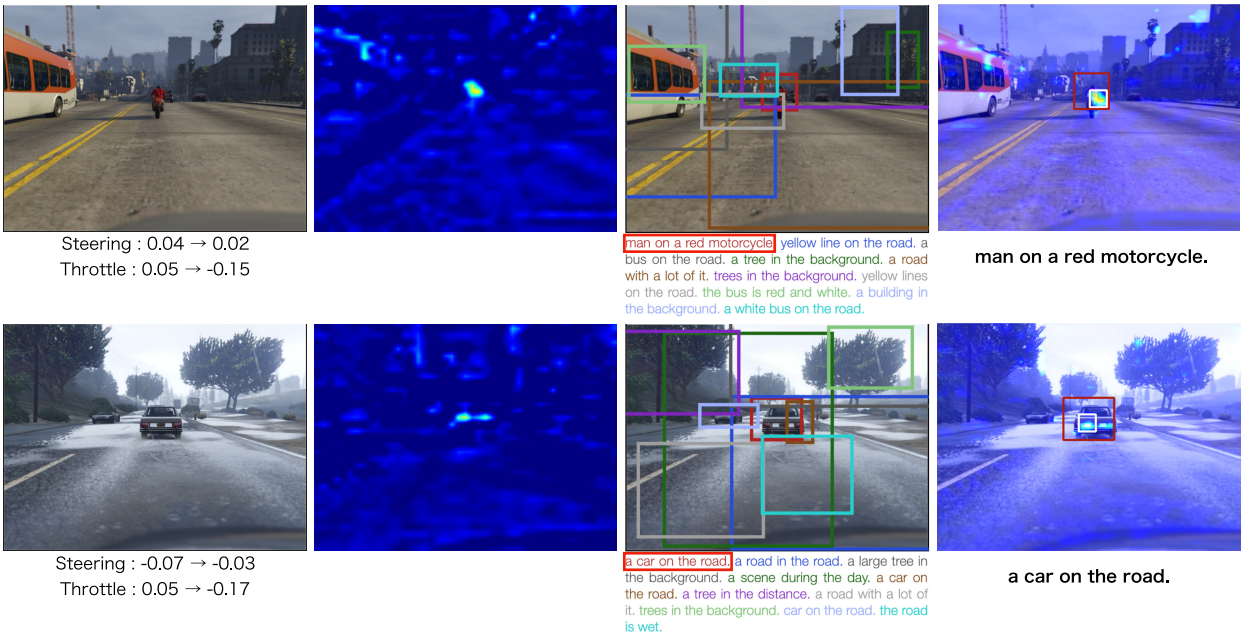


Fig. 5. Examples of the selected captions from the dense captioning with an FCLN by using the peak in an attention map on self-driving. (a) input image and the estimated steering and throttle, (b) the corresponding attention maps, (c) captions obtained from an FCLN, and (d) the selected captions from the results of an FCLN by using the attention map.

attention map can explain the reason of the decision making linguistically.

V. CONCLUSIONS

In this paper, we proposed an end-to-end self-driving method that estimates steering and throttle and visually explained the reason of the network output for the purpose of complete self-driving. By using a velocity as an additional input, the proposed method improves the control performances of steering and throttle. And, the introduced ABN structure generates attention maps for visual explanation. As an application of the obtained attention maps, we combined the attention map and an FCLN that generates captions for specific regions. The generated captions are akin to the possible reason of the decision making. Our future work includes developing an interactive and linguistic explanation method of the decision making on the basis of the proposed method.

REFERENCES

[1] Q. Li, L. Chen, M. Li, S. Shaw, and A. Nüchter, "A sensor-fusion drivable-region and lane-detection system for autonomous vehicle navigation in challenging road scenarios," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 2, pp. 540–555, Feb 2014.

[2] U. Lee, J. Jung, S. Jung, and D. H. Shim, "Development of a self-driving car that can handle the adverse weather," *International Journal of Automotive Technology*, vol. 19, no. 1, pp. 191–197, Feb 2018.

[3] 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," *arXiv preprint, arXiv:1604.07316*, 2016.

[4] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-End Learning of Driving Models from Large-Scale Video Datasets," in *Computer Vision and Pattern Recognition*, 2017, pp. 3530–3538.

[5] J. Kim and J. Canny, "Interpretable Learning for Self-Driving Cars by Visualizing Causal Attention," in *International Conference on Computer Vision*, 2017, pp. 2961–2969.

[6] L. Yang, X. Liang, T. Wang, and E. Xing, "Real-to-Virtual Domain Unification for End-to-End Autonomous Driving," in *European Conference on Computer Vision*, 2018, pp. 553–570.

[7] H. Fukui, T. Hirakawa, T. Yamashita, and H. Fujiyoshi, "Attention Branch Network: Learning of Attention Mechanism for Visual Explanation," *arXiv preprint arXiv:1812.10025*, 2018.

[8] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conference on Computer Vision*, 2014, pp. 818–833.

[9] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning Deep Features for Discriminative Localization," in *Computer Vision and Pattern Recognition*, 2016, pp. 2921–2929.

[10] D. Smilkov, N. Thorat, B. Kim, F. B. Viégas, and M. Wattenberg, "SmoothGrad: removing noise by adding noise," *arXiv preprint, arXiv:1706.03825*, 2017.

[11] 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 *International Conference on Computer Vision*, 2017, pp. 618–626.

[12] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks," in *Winter Conference on Applications of Computer Vision*, 2018, pp. 839–847.

[13] M. Lin, Q. Chen, and S. Yan, "Network In Network," in *International Conference on Learning Representations*, 2014.

[14] M. Bojarski, P. Yeres, A. Choromanska, K. Choromanski, B. Firner, L. D. Jackel, and U. Muller, "Explaining How a Deep Neural Network Trained with End-to-End Learning Steers a Car," *arXiv preprint, arXiv:1704.07911*, 2017.

[15] M. Bojarski, A. Choromanska, K. Choromanski, B. Firner, L. D. Jackel, U. Muller, and K. Zieba, "VisualBackProp: visualizing CNNs for autonomous driving," *arXiv preprint, arXiv:1611.05418*, 2016.

[16] J. Johnson, A. Karpathy, and L. Fei-Fei, "DenseCap: Fully Convolutional Localization Networks for Dense Captioning," in *Computer Vision and Pattern Recognition*, 2016, pp. 4565–4574.