# Intelligent Driving Data Recorder in Smartphone Using Deep Neural Network-Based Speedometer and Scene Understanding

Yanlei Gu, *Member, IEEE*, Qianlong Wang, and Shunsuke Kamijo, *Senior Member, IEEE*

***Abstract*— This paper proposes a smartphone-based Driving Data Recorder (DDR). The proposed DDR has the functions of accurate speed estimation and intelligent traffic scene understanding. DDRs are used to store the relevant driving data to provide feedback on driver behavior for accident analysis, insurance issue, and so on. The conventional DDRs are standalone devices with multiple sensors. The current DDR products record many useless data or lose important information. On the other hand, the widely used smartphones already have the hardware conditions to replace the conventional DDR products. This paper proposes to develop the intelligent DDR in the smartphones. Considering the requirements of the DDRs, two functions are developed in this paper: motion sensor-based speedometer and vision sensor-based scene understanding. The proposed speedometer function adopts double-layered Long Short-Term Memory (LSTM) network as the model, which can estimate the vehicle speed directly from gyroscope and accelerometer of a smartphone. The scene understanding function can detect road facilities such as traffic lights, crosswalks, and stop lines. The driving data recorded in those areas are very important for analyzing driver behaviors. In the development of the scene understanding function, maintaining high detection accuracy with reduced computation cost is significant due to the limitation of smartphones' processing resources. This paper uses a lightweight architecture deep learning network to achieve the goal. The proposed system has been evaluated using the real traffic data. Speed estimation function only has 1.8 km/h of speed mean error. In addition, there is no accumulated error even for a long time driving. The evaluation of the scene understanding function indicates that the proposed method can provide a high-accuracy detection at 2 FPS, which is faster than the state-of-the-art method.**

***Index Terms*— Driving data recorder, smartphone, deep learning, speedometer, scene understanding, LSTM, CNN compression.**

## I. Introduction

THE driving data recording systems are used to store the relevant driving data and to provide feedback on driver behavior [1], [2]. These devices provide data that so helps identify driver maneuvers associated with risky driving and overall trip safety objectively [3]. Current driving data recorders (DDRs) in the market are the standalone devices with Global Navigation Satellite System (GNSS) receiver, accelerometer, and camera. Some DDRs are activated by crash-like events (such as sudden changes in velocity) and may continue to record until the accident is over, or until the recording time is expired. An investigation by Japan Trucking Association [4] indicates the problems of the current DDRs in the market. 43.9% of the customers pointed out the high cost problem of the DDRs. Also, 36.4% of the investigated participants gave the comments that many useless data are recorded by the DDRs. Thus, it is difficult to search the important data from the huge recorded data. The more intelligent and cheaper DDR is expected by the automotive owners.

Smartphones have become essential devices in the lives of the public. The smartphone ownership rate is increased to 77% among the adults in United States [5]. It is economical and convenient for drivers if the DDR could be developed in the smartphones. Because drivers do not need to buy the additional DDR device and manage it. In addition, the mainstream products of smartphones have contained the necessary sensors for developing DDRs, such as camera, GPS receiver, accelerometer and gyroscope [6]. Moreover, the newest generation of smartphones also increases the computing power of CPU and even includes mobile version GPU to improve the image processing ability [7]. It is clear to see that the smartphone could be the appropriate platform for developing the intelligent DDR.

Smartphone sensors have been used for the research of mining driving routes [8], traffic monitoring and recognition of the aggressive driving behaviors. In an early stage, Mohan *et al.* [9] proposed to use the accelerometer and Global Positioning System (GPS) sensor in the smartphones for detecting potholes, bumps as well as vehicles braking and honking, and reporting the relevant location. Estimation of road conditions using smartphone accelerometer and gyroscope has been proposed by Allouch *et al.* [10] as well. Thiagarajan *et al.* [11] proposed a system called vTrack. This system monitors the traffic and predicts the traveling time for users based on the report from GPS and WiFi position sensors in massive users' smartphones. Recently, Johnson and Trivedi [12] proposed an approach in order to

Manuscript received August 8, 2018; accepted October 3, 2018. Date of publication October 9, 2018; date of current version December 7, 2018. This paper was presented at the 2018 IEEE International Conference on Consumer Electronics and was published in its Proceedings (https://ieeexplore.ieee.org/document/8326122/). The associate editor coordinating the review of this paper and approving it for publication was Dr. Ravibabu Mulaveesala. *(Corresponding author: Yanlei Gu.)*

Y. Gu is with the Institute of Industrial Science, The University of Tokyo, Tokyo 153-8505, Japan (e-mail: guyanlei@kmj.iis.u-tokyo.ac.jp).

Q. Wang is with the Graduate School of Interdisciplinary Information Studies, The University of Tokyo, Tokyo 153-8505, Japan (e-mail: wangqianlong@kmj.iis.u-tokyo.ac.jp).

S. Kamijo is with the Interfaculty Initiative in Information Studies, The University of Tokyo, Tokyo 153-8505, Japan (e-mail: kamijo@iis.u-tokyo.ac.jp).

Digital Object Identifier 10.1109/JSEN.2018.2874665

classify different driving styles based on data collected from smartphones . The proposed approach can classify the driving styles into the form of normal, aggressive and very aggressive based on the rotation rate and acceleration. Furthermore, Saiprasert *et al.* [6] used motion sensors (GPS receiver and accelerometer) embedded on a smartphone to record and detect 12 types of driving events, for example, sudden lane changing or sudden turning. They proposed pattern matching algorithm to detect driving events based on the use of DTW algorithm.

Most of the researchers considered the position, acceleration and angle rate information in the traffic monitoring and driving behavior recognition. However, the speed information is also significant for analyzing the driving behaviors and events. Firstly, road speed limits are legally used in most countries. Drivers receive the penalty for the violation of speed limits. Moreover, research in the accidents indicated that the risk of being involved in an accident as a result of excess speed increases [3]. Thus, the speed information of vehicle should be considered in the development of the DDRs.

There are several ways to obtain the speed information. The first way is acquiring the speed from vehicle Control Area Network (CAN). However, accessing to CAN data needs to develop the specific interface hardware, and the compatibility for the format of the CAN data should be considered for the different automobile brands. The second method is to estimate the vehicle speed for GNSS data. However, multipath reflection and blockage of GPS signal occur in urban areas, which decrease the accuracy of the speed estimation [13]. More seriously, GNSS signal itself can be lost in certain areas, e.g. in the city with tall buildings or under bridges [6], [14]. Therefore, the GNSS should be reconsidered or excluded for the speed estimation. At least, other methods should be developed as the supplement.

Besides CAN and GNSS based methods, the speed value can be estimated by from the accelerometer, theoretically. Fazeen *et al.* [15] obtained vehicle speed by integrating the data from accelerometer using the trapezoidal method. Authors compared the estimated speed with the reference speed recorded from the car' s dashboard for the evaluation. The result shows that this method is accurate at low speeds and short distance. But the error accumulation may happen for the long distance case. Yu *et al.* [16] applied integrating method to estimate the speed as well. But, they proposed to reset the estimated speed when such situations happen, e.g. making turns, stopping, and passing over uneven road surfaces. Their proposed system utilizes two sensors in the smartphone: accelerometer and gyroscope. The accelerometer and gyroscope were integrated for detecting reference situations and estimating the instant vehicle speed. But the method is not robust enough. When running on a path without reference points, the accurate result is not achievable by their method.

Current DDRs record many useless data. One reason of this problem is that the DDRs usually do not include the online image analysis function. The rich information contained in images can explain the surrounding environment. By understanding the surrounding environment and associating with the speed and acceleration information, DDRs can focus on the specific scenarios and critical areas where the driving data are significant for evaluating the driver behavior. For example, with the vision based scene understanding function, it is possible to detect the status of the traffic light and the position of the stop line at the intersection. The recording function can be triggered based on the location and traffic situation, also the recorded data can be automatically categorized. The advantage of the scene understanding function is attractive, but developing such function in a portable device is also challenging because of the limitation of the computation power in portable devices.

Vision based object detection is a popular topic. The detection of road facilities and objects in traffic scene has been discussed intensively with the development of the autonomous driving. In the early stage, researcher adopted the shallow methods for object detection tasks [17]–[25]. The shallow methods mainly rely on human designed features for building the detection system. However, handcrafted features have recently been significantly outperformed by other types of methods based on deep learning.

Convolutional neural network (CNN) is one of the typical deep learning architectures. It a special kind of multi-layer neural network designed to recognize visual patterns directly from pixel images with minimal preprocessing. CNN is now considered as the most effective method for the image classification tasks. LeNet the first successful application of CNN to digit recognition which was developed by Yann LeCun in 1990s [26]. LeNet consists of a sequence of convolutional, max pooling layers followed by a fully connected layer. The recent progress of CNN started from 2012, in which AlexNet significantly outperformed all the prior competitors and won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [27]. After that, GoogleNet proposed from Google won the ILSVRC 2014 competition [28]. ResNet won the ILSVRC 2015 [29].

Those CNNs are developed for the classification tasks. The detection task is different with classification. The detection algorithm should provide the position of the object from the complex background. Region-Based Convolutional Network (R-CNN) is the prior CNN based detection network. R-CNN used a selective search algorithm to extract around 2000 bottom-up region proposals, each region is passed into a large classification CNN [30]. After R-CNN, Fast R-CNN and Faster R-CNN were successively proposed to reduce the computation cost [31], [32]. But these methods still suffer from the problem of the processing time, especially for the portable device. The other idea is to consider the detection task as a single regression problem, where bounding box coordinates and class probabilities are computed at the same time. You Only Look Once (YOLO) is the representative method [33], [34]. After YOLO, a Single Shot multibox Detector (SSD) was proposed. Different from YOLO, SSD discretizes the output space of bounding boxes into a set of default boxes over different aspect ratios and scales to better match the object shape. Additionally, the network combines predictions from multiple feature maps with different resolutions to naturally handle objects of various sizes [35]. However, according to our experience, the well-known CNN based detection systems (e.g. Faster-RCNN, YOLO, and SSD) are all unable to achieve an acceptable processing time on the smartphone.

Based on the comparison with the related works and our proposed system, three contributions of this paper can be summarized as follows. Firstly, this paper proposes a deep learning based speed estimation method. The proposed method can accurately estimate the vehicle speed from the data of accelerometer and gyroscope, and does not have the error accumulation problem even for the long and straight road test. Secondly, this paper proposes and designs a lightweight architecture CNN in the smartphone for the vision based scene understanding. The developed CNN can recognize the important road facilities with an acceptable processing time in the smartphone. Thirdly, the scenario based recording function is discussed for the future practical application in this paper. The initial ideas of the smartphone based DDRs have published in our previous conference papers [36], [37]. Compared to our previous conference papers, this paper gives more surveys about the related works, and also provides the full description of the proposed ideas. In addition, this paper discusses the scenario based recording function.

The rest paper is organized as follows: Section II descripts the deep learning based speed estimation method. Section III explains the proposed CNN for traffic scene understanding. Section IV evaluates the proposed two ideas in Section II and III, and also discusses the scenario based recording function. Finally, Section V concludes this paper.

## II. Speed Estimation from Gyroscope and Accelerometer

### A. LSTM Network

Long Short-Term Memory (LSTM) network was firstly proposed by Hochreiter and Schmidhuber [38]. LSTM networks are recurrent neural networks equipped with a special gating mechanism that controls access to memory cells. Since the function of the gates, LSTM and its variant have recently shown great promise in tackling various sequence modeling tasks in machine learning, e.g. natural language processing [39], image captioning [40] and speech recognition [41].

Basically, a LSTM unit consists of an input gate, a forget gate, and an output gate. Suppose that $\mathbf{x}_t$ is the input, $\mathbf{h}_{t-1}$ is the hidden output from the last time step $t-1$, the input gate decides how much the new information will be added to the cell state $\mathbf{c}_t$, and generates a candidate $\tilde{\mathbf{c}}_t$ by:

$$\mathbf{i}_t = \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i) \tag{1}$$

$$\tilde{\mathbf{c}}_t = \phi(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c) \tag{2}$$

where, $\mathbf{i}_t$ can be thought of as a knob that the LSTM learns to selectively consider $\tilde{\mathbf{c}}_t$ for the current time step. $\sigma$ is the logistic sigmoid function, $\phi$ is $tanh$ in this paper. Generally, $\mathbf{W}$ terms denote weight matrices (e.g. $\mathbf{W}_{xi}$ is the matrix of weights from the input to the input gate), and $\mathbf{b}$ terms are the bias vectors. The forget gate decides how previous information will be kept in the new time step, and is defined as:

$$\mathbf{f}_t = \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f) \tag{3}$$

Then, the cell state $\mathbf{c}_t$ will be updated by:

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \tag{4}$$

where $\odot$ is the element-wise product of the vectors. Then, the output gate uses the output $\mathbf{o}_t$ to control what is then read from the new cell state $\mathbf{c}_t$ onto the hidden vector $\mathbf{h}_t$ as follows:

$$\mathbf{o}_t = \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o) \tag{5}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \phi(\mathbf{c}_t) \tag{6}$$

This paper uses the functional LSTM(·,·,·) as shorthand for the LSTM model in equation (7):

$$(\mathbf{h}_t, \mathbf{c}_t) = \mathrm{LSTM}\,(\mathbf{x}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}, \mathbf{W}, \mathbf{b}) \tag{7}$$

where $\mathbf{W}$ and $\mathbf{b}$ include the weight matrices and bias vectors indicated in equations (1) to (5). The value of $\mathbf{W}$ and $\mathbf{b}$ are determined in the training step.

### B. Concatenated Double-Layered LSTM for Speed Estimation

This paper proposes a LSTM based network to estimate the vehicle speed from the sequence data recorded by the accelerometer and gyroscope of a smartphone. There are two points are considered in the design of the network. The first one is how to build the correspondence between the sequence data $(\mathbf{r}_1, \ldots, \mathbf{r}_t)$ and estimated $speed_t$ by means of regarding time $t$ in the macroscopic level. The second point is how to optimize the model in microscopic level. Both two considerations focus on obtaining the accurate speed.

"Many to one" model has been widely used in the sequence data processing. In order to fully use the memory and forget ability of LSTM, our proposed network is also a "many to one" model in macroscopic level. The architecture of the proposed speed estimation network is shown in Figure 1. It means when estimating the $speed_t$, the sensor data from time $t-n$ to $t$: $(\mathbf{r}_{t-n}, \ldots, \mathbf{r}_t)$ are input to the network together with the hidden state $\left(\mathbf{s}^1_{t-n-1}, \mathbf{s}^2_{t-n-1}\right)$ of the network. The reason of using "many to one" is that the speed estimation is not a simply integral operation from acceleration data in our network. The mechanism of correcting the accumulated error is expected to be built into the network. "Many to one" model uses the information in a longer time period than "one to one" model. Thus, "many to one" model provides more chances to acquire the useful information from the sequence data for the error correction. In the "many to one" network, the parameters in different models (Model 1 ... Model $n$ in Figure 1) actually are different. The different parameters also reflect the time related influence from the input sequence data to the speed output. In the proposed network, $n$ is empirically decided as 40 by considering both the accuracy and computation cost. Since the frequency of the sensor data is 10Hz, $n = 40$ means that our system needs previous 4-second sensor data to estimate the current vehicle speed.

Figure 2 is corresponding to Model $n$ in Figure 1, and shows the architecture of our network in the microscopic level. There are two hidden layers and double-layered LSTMs. Finally, the output is connected with the last hidden layer. Here, $\mathbf{r}_t$ denotes

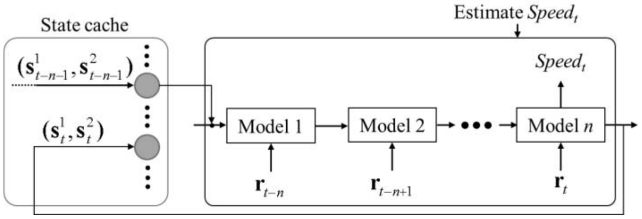


Fig. 1. "Many to one" architecture for speed estimation using data sequence.

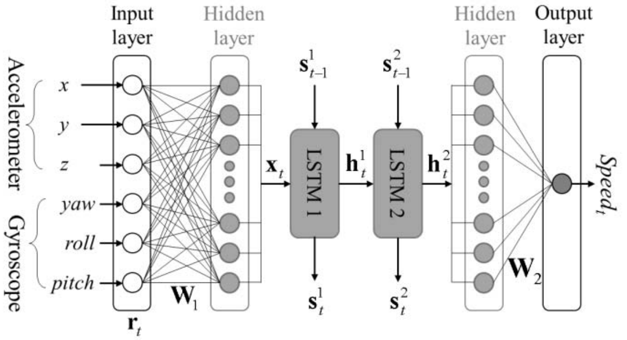


Fig. 2. Architecture of double-layer LSTM model for speed estimation.

the raw data from the accelerometer and gyroscope in time $t$. The output of the first hidden layer is:

$$\mathbf{x}_t = \mathbf{W}_1\mathbf{r}_t + \mathbf{b}_1 \tag{8}$$

where, $\mathbf{x}_t$ is a vector, it stands for the features extracted from raw data $\mathbf{r}_t$ in time $t$. $\mathbf{W}_1$ and $\mathbf{b}_1$ are the weight matrix and bias from the input layer to the first hidden layer, respectively. In our proposed network, the number of the features in the first hidden is empirically decided. This paper extracts 128 features from the input layer, so the length of the vector $\mathbf{x}_t$ is 128. The dimension of the feature is much higher than the dimension of the raw data. The reason of this process is to give enough input to LSTM unit and make LSTM fully perform its function of choosing the useful information.

In addition, inspired by [42] and [43], this paper adopts a double-layered LSTM model to represent the hierarchical structure of sensor data sequences, as shown in Figure 2. Comparing with one-layered LSTM and three-layered LSTM, double-layered LSTM model provides a better result in our experiments. The double-layered LSTM model can be explained by:

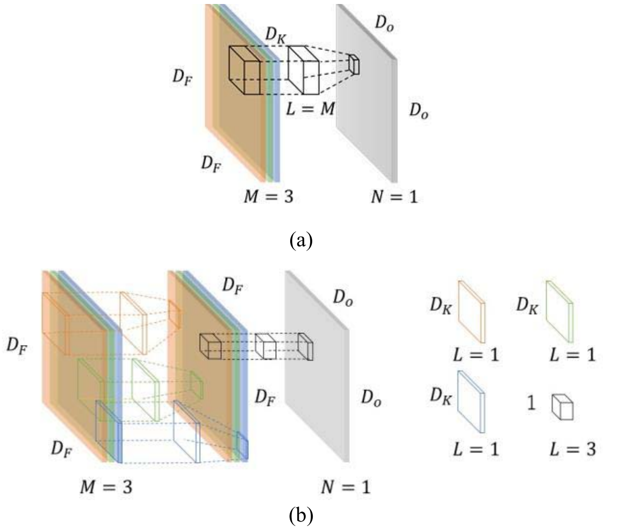
$$\left(\mathbf{h}^1_t, \mathbf{c}^1_t\right) = \text{LSTM 1}\left(\mathbf{x}_t, \mathbf{h}^1_{t-1}, \mathbf{c}^1_{t-1}, \mathbf{W}^1, \mathbf{b}^1\right) \tag{9}$$

$$\left(\mathbf{h}^2_t, \mathbf{c}^2_t\right) = \text{LSTM 2}\left(\mathbf{h}^1_t, \mathbf{h}^2_{t-1}, \mathbf{c}^2_{t-1}, \mathbf{W}^2, \mathbf{b}^2\right) \tag{10}$$

In Figure 2, $\mathbf{s}^n_t$ stands for $\left(\mathbf{h}^n_t, \mathbf{c}^n_t\right), n = 1$ or 2. Finally, the speed can be estimated from the output of the second LSTM layer as:

$$speed_t = \mathbf{W}_2\mathbf{h}^2_t + \mathbf{b}_2 \tag{11}$$

where $\mathbf{W}_2$ is a weight matrix from the hidden layer to the output layer, and $\mathbf{b}_2$ is the bias vector.



(a)

(b)

Fig. 3. (a) Standard convolution. (b) Depth-wise separable convolution.

In the training of the network, this research uses $\left\{(\mathbf{r}_{t-n}, \ldots, \mathbf{r}_t)\,; speed^{ground\ truth}_t\right\}$ as the sample data because of the "many to one" architecture. The Adadelta training algorithm [44] automatically adjusts the parameters in $\{\text{Model } 1, \ldots, \text{Model } n\}$ based on the principle of the gradient descent. The result of the speed estimation will be detailed described in the Section IV.

## III. Road Facility Detection

### A. Deep CNN Compression

Object detection is essential for traffic scene understanding. However, according to our experience, the well-known deep CNN based detection algorithms (e.g. Faster-RCNN [32], YOLO [33], SSD [35]) are unable to achieve acceptable processing time on the smartphone. This paper designs a much less computational detection network that is able to achieve high accuracy as well.

A standard convolution both filters and combines inputs into a new set of outputs in one step, as shown in Figure 3 (a). The depth-wise separable convolution is an effective way to reduce the computation cost. The idea of depth-wise separable convolution was proposed by Howard *et al.* [45] in the MobileNet. It splits the standard convolution into two operations, one operation for filtering and the other one for combining, as shown in Figure 3 (b).

In Figure 3, $D_F$ denotes the size of the input feature map, $D_K$ is the kernel size of filters, and $D_O$ is the size of the output feature map. $L$ is the depth of the kernel. $M$ and $N$ are the numbers of the input and output channels, respectively. This paper sets $M = 3$ and $N = 1$ in Figure 3 in order to illustrate the depth-wise separable convolution. In this example of Figure 3 (a), the computational complexity of standard convolution is:

$$D_F \times D_F \times M \times D_K \times D_K \times N \tag{12}$$

Depth-wise separable convolution is made up of two parts: depth-wise convolution and pointwise convolution. The

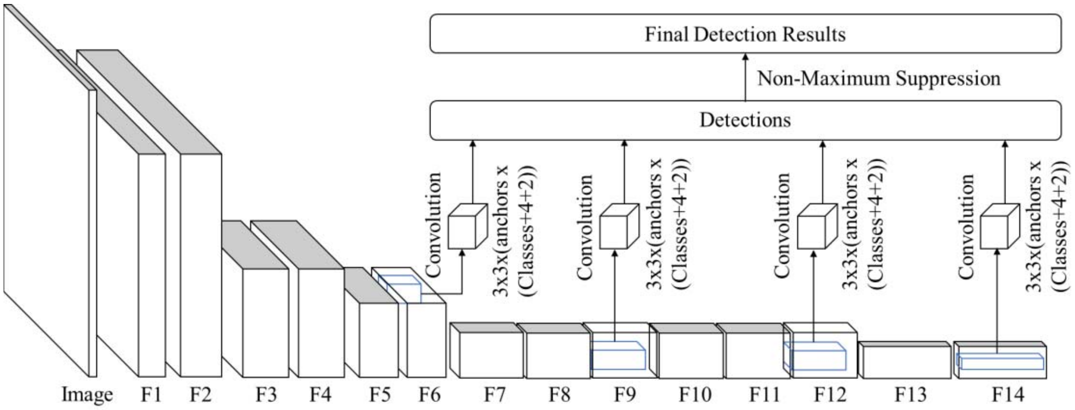


Fig. 4. Architecture of the proposed road facility detection network.

depth-wise convolution applies a single filter to each input channel, so the number of channels will not change after depth wise convolution. After that, pointwise convolution, a 1 × 1 × 3 convolution is used to combine the output of depth-wise convolution. The computational complexity of depth-wise separable convolution in Figure 3 (b) can be formulated as:

$$D_F \times D_F \times M \times D_K \times D_K + M \times D_F \times D_F \times N \quad (13)$$

$$\frac{D_F \times D_F \times M \times D_K \times D_K + M \times D_F \times D_F \times N}{D_F \times D_F \times M \times D_K \times D_K \times N} = \frac{1}{N} + \frac{1}{D_K^2} \quad (14)$$

As calculated in equation (14), the computation complexity of depth-wise separable convolution is $\frac{1}{N} + \frac{1}{D_K^2}$ of the standard convolution. This paper employs the depth-wise separable convolution in the developed road facility detection network.

### *B. Light and Multiscale Road Facility Detection Network*

Figure 4 visualizes the architecture of our proposed road facility detection network. The primary structure is mainly inspired by SSD [35]. In addition, this paper adopts the depth-wise separable convolution instead of the standard convolution in our proposed network. In fact, the combination of SSD and MobileNet has been mentioned by Howard *et al.* [45], but the detail of the developed network is not disclosed. In this research, we design and implement the network for our application, make the necessary optimization for the good performance.

The proposed road facility detection network is a feed-forward convolutional network that produces a fixed-size collection of bounding boxes for the detection in different layers and scores for the presence of object class in those boxes, then followed by a non-maximum suppression step to generate the final detection results, as shown in Figure 4. Table I gives the details of the convolution operations described in Figure 4.

In the design of the road facility detection network, two problems should be considered. Firstly, the sizes of detected objects differ a lot in traffic scene, e. g. the traffic light is very small, but the crosswalk area is relatively big, as shown in Figure 5. As for this problem, this paper applies multiscale feature maps for the detection. In the proposed network, the shallow layers (F6, F9 in Figure 4) is used to detect traffic lights and other deeper layers (F12, F14 in Figure. 4) are utilized to detect crosswalk targets with the big size. Secondly, the shapes of the road facilities are so different, e. g. the stop is a horizontal bar and more narrow than the traffic light rectangle as shown in Figure 5. This paper borrows the idea of SSD and defines the several default boxes with different aspect ratios for detecting the different shapes of objects together. Allowing

TABLE I

DETAILS OF CONVOLUTION OPERATIONS IN THE PROPOSED ROAD FACILITY DETECTION NETWORK

| Operation | Output Feature Map |
|---|---|
| Conv-k3-s2-n32 | F1: 220×220×32 |
| DepthSepConv-k3-s1-n64 | F2: 220×220×64 |
| DepthSepConv-k3-s2-n128 | F3: 110×110×128 |
| DepthSepConv-k3-s1-n128 | F4: 110×110×128 |
| DepthSepConv-k3-s2-n256 | F5: 55×55×256 |
| DepthSepConv-k3-s1-n256 | F6: 55×55×256 |
| DepthSepConv-k3-s2-n512 | F7: 28×28×512 |
| 2×DepthSepConv-k3-s1-n512 | F9: 28×28×512 |
| 3×DepthSepConv-k3-s1-n512 | F12: 28×28×512 |
| DepthSepConv-k3-s2-n1024 | F13: 14×14×1024 |
| DepthSepConv-k3-s1-n1024 | F14: 14×14×1024 |

E.g. k3: kernel size is 3x3; s2: convolution stride is 2; n64: The number of the output feature maps is 64.

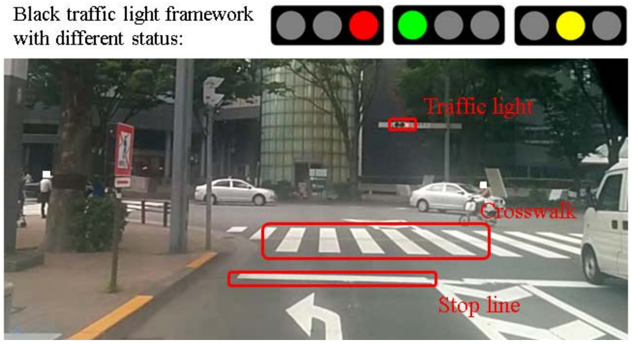


Fig. 5. Road facilities in image from smartphone camera.

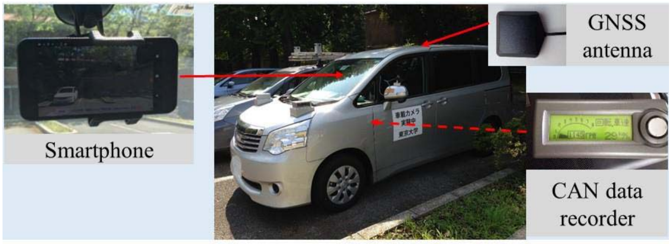


Fig. 6. Configuration of smartphone and other devices in experiments.

different default box shapes in several scale feature maps let the network efficiently discretize the space of possible output box shapes [35].

In addition to the multiscale and default boxes, this paper introduces a new loss term for improving the recognition accuracy of the traffic light. In the development of the intelligent DDR, it is very significant to recognize the state of the traffic light, because the meaning of the driving behavior should be understood by correlating with the state of the traffic light. However, traffic light is the very small object, but the framework is also included in the device of the traffic light, as shown in Figure 6. The proposed network firstly utilizes the light area and framework area for traffic light detection from background, and then focus on the light color in the classification of the traffic light state. This point is considered in the definition of the loss function. Here, this paper just explains the difference between the original SSD and our network, the explanation of the similar parts could be found in the paper of SSD [33]. Equations (15) and (16) show the overall objective loss function in the original SSD and our network, respectively.

$$L(x,c,l,g) = \frac{1}{N}\left(L_{conf}(x,c) + \alpha L_{loc}(x,l,g)\right) \quad (15)$$

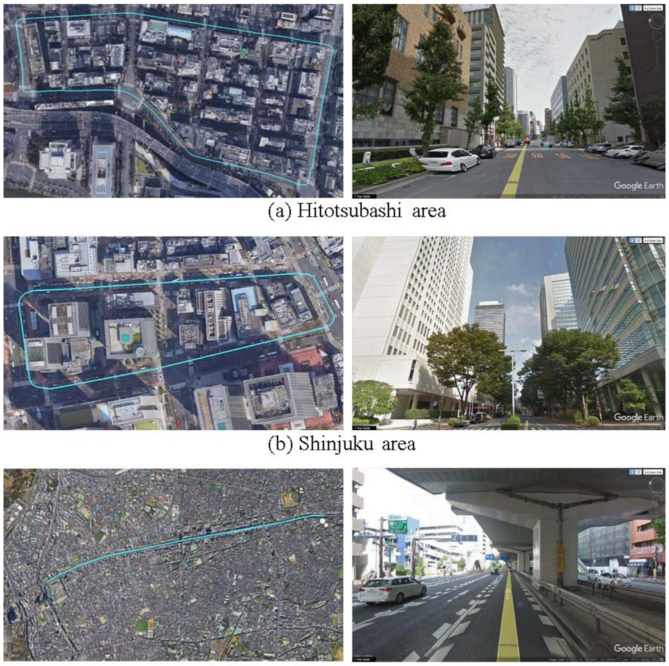
$$L(x,c,o,l,g) = \frac{1}{N}\Big(L_{class}(x,c) + L_{obj}(x,o) + \alpha L_{loc}(x,l,g)\Big) \quad (16)$$

where $N$ is the number of matched default boxes. This paper sets loss to 0 when $N = 0$. The localization loss $L_{loc}(x,l,g)$ is the same as the one in the loss function of SSD. But the confidence loss $L_{conf}(x,c)$ is extended to two part $L_{class}(x,c)$ and $L_{obj}(x,o)$ as:

$$L_{class}(x,c) = -\sum_{i\in Pos}^{N} x_{ij}^{p}\log\left(\hat{c}_i^{p}\right) \quad where \quad \hat{c}_i^{p} = \frac{\exp\left(c_i^{p}\right)}{\sum_p \exp\left(c_i^{p}\right)} \quad (17)$$

$$L_{obj}(x,o) = -\sum_{i\in Pos}^{N} x_{ij}^{1}\log\left(\hat{o}_i^{1}\right) - \sum_{i\in Neg}\log\left(\hat{o}_i^{0}\right) \quad where \quad \hat{o}_i^{p} = \frac{\exp\left(o_i^{p}\right)}{\sum_p \exp\left(o_i^{p}\right)} \quad (18)$$

where, the term $L_{obj}(x,o)$ is used to detect road facilities from the background, and $L_{class}(x,c)$ can be used to recognize the difference among the road facilities. We expect the small difference between the different color traffic lights can be recognized using the term $L_{class}(x,c)$.



(a) Hitotsubashi area

(b) Shinjuku area

(c) Normal road under viaduct

Fig. 7. Test routes and typical scenes around routes. (a) Hitotsubashi area. (b) Shinjuku area. (c) Normal road under viaduct.

## IV. Experimental Results

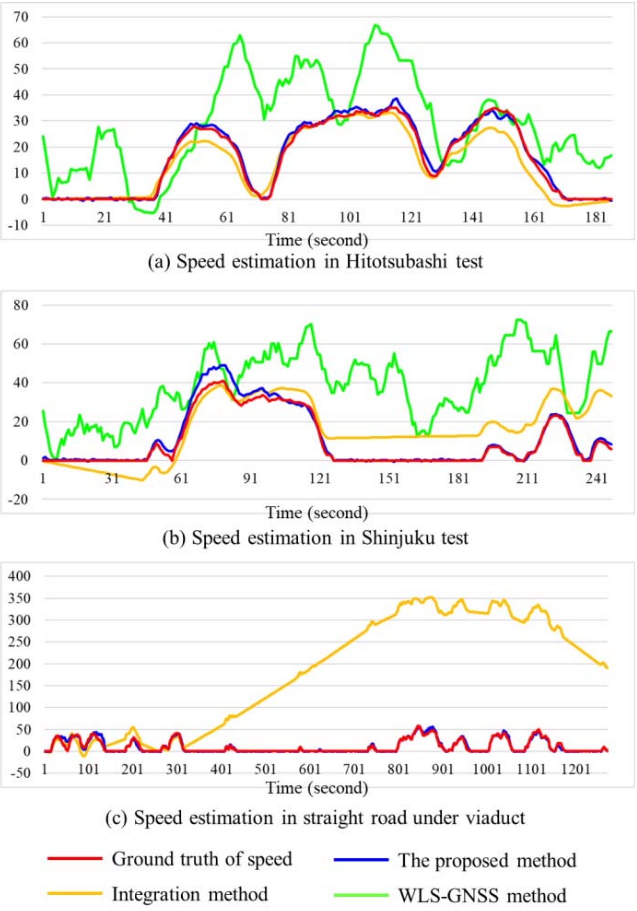
In order to verify the performance of the proposed method, we collected driving data by ourselves when we were driving in Tokyo. Figure 6 shows the configuration of the smartphone in our experimental vehicles. This research uses motion sensors embedded on the smartphone (Google Pixel) to collect the acceleration and angular velocity, the images are collected from smartphone camera. In addition, the GNSS receiver and CAN data recorder are also equipped on the experimental vehicles for the evaluation. The CAN data recorder is connected with CAN interface of vehicle to record the speed value as the ground truth.

### A. Evaluation of Speed Estimation Function

In the evaluation for the speed estimation function, this research used about 50-hour driving data, including 45-hour data recorded in the urban area and 5-hour data recorded under the National Highway. The data is divided into two different parts: training data and test data. This research uses the data at three different locations for the test. The three locations are Hitotsubashi Area, Shinjuku Area, and normal road under the viaduct of National Highway in Tokyo. The routes and the typical scenes around the routes are shown in Figure 7. The light blue lines indicate the test routes. Hitotsubashi area and Shinjuku area are typical urban environments and have many skyscrapers around roads. The test routes are the circle with several turning at the Hitotsubashi area and Shinjuku area. At the same time, this research also

TABLE II
COMPARISON BETWEEN DIFFERENT SPEED ESTIMATION METHODS

| Test area | Method | Speed error mean (km/h) |
|---|---|---|
| Hitotsubashi area | Our Method | 1.61 |
| | Integration Method | 9.74 |
| | WLS-GNSS Method | 28.78 |
| Shinjuku area | Our Method | 1.80 |
| | Integration Method | 6.57 |
| | WLS-GNSS Method | 15.68 |
| Normal road under viaduct | Our Method | 2.23 |
| | Integration Method | 170.34 |



(a) Speed estimation in Hitotsubashi test

(b) Speed estimation in Shinjuku test

(c) Speed estimation in straight road under viaduct

Fig. 8. Parts of speed estimation results in urban areas. (a) Speed estimation in Hitotsubashi test. (b) Speed estimation in Shinjuku test. (c) Speed estimation in straight road under viaduct.

selects a straight path for testing our proposed speed estimation network, as shown in Figure 7 (c). There is no sharp turning and the length of the route is also longer. As shown in the right image of Figure 7 (c), the driving route is covered by the viaduct of National Highway. The total driving distance of the three routes are about 5 km, 10 km, and 20 km.

The performance of our purposed speed estimation network is summarized in Table II. In order to understand the advantage of our proposed idea, two conventional methods are used for the comparison. The first one is the GNSS based speed estimation. The speed is calculated from consecutive epochs estimated from the widely used Weighted Least Squares GNSS (WLS-GNSS) positioning method. The second method is the integration method proposed by Fazeen *et al.* [15]. We implemented this integration method for the comparison. The speed error mean is used for the evaluation as:

$$Speed\ Error\ Mean = \frac{\sum_{t=1\ldots N} \left| speed_t^e - speed_t^g \right|}{N} \quad (19)$$

where, the $speed_t^e$ is the estimated speed at time $t$, $speed_t^g$ is the speed ground truth at time $t$. $N$ is the time length of the data. In this research, the output speed from the CAN is considered as the speed ground truth. As shown in Table II, in the test of circle routes at Hitotsubashi area and Shinjuku area, our proposed method achieved 1.6 km/h and 1.8 km/h of speed mean error. In the test on the straight and long path, the proposed method also can provide 2.3 km/h speed mean error. As indicated in Table II, the proposed method is much more accurate than the WLS-GNSS method and the integration method.

Figure 8 visualizes some parts of the speed estimation results in different tests performed in the urban areas. Red curve indicates the ground truth of the vehicle speed, green curve indicates the result of the WLS-GNSS method, yellow curve is the vehicle speed estimated by the integration method and the blue curve is the speed estimated from our proposed method. Because the driving route is covered by the viaduct of National Highway and tall buildings are on the side of the road, GNSS receiver cannot receive enough GNSS signal for positioning. Therefore, WLS-GNSS method is not compared in this case for Table II and Figure 8. Obviously, the WLS-GNSS method has the huge error in Hitotsubashi and Shinjuku tests. The reason is that the quality of GNSS signal is degraded in the central business district of a city because of the signal blockage and reflection [46]. Also, the integration method shows the error accumulation problem, especially for the long distance test on the straight route. The speed estimated from our proposed method is almost the same as the ground truth, and there is no accumulated error. In addition, the training data does not include any data collected on the route of Shinkuku area, and our proposed method still can provide the accurate speed estimation result. It proves that the proposed method can be applied into other places without increasing the corresponding training data.

### *B. Evaluation of Road Facility Detection*

This paper focuses on the road facilities of intersection areas: traffic light, stop line and crosswalk. In order to develop the road facility detection function, we label 70 image sequences for training and test. The data is collected from different places and intersections. 50 sequences are used for training, and other 20 sequences are the test data. In the evaluation of the accuracy of the proposed detection network, this research adopts the average precision indicator and follows the PASCAL protocol [47]. In addition, when overlap region between detection result and ground truth is over than 50%, the detection is considered as correct one.

Table III shows the performance of our proposed road facility detection network, and Figure 9 gives some examples

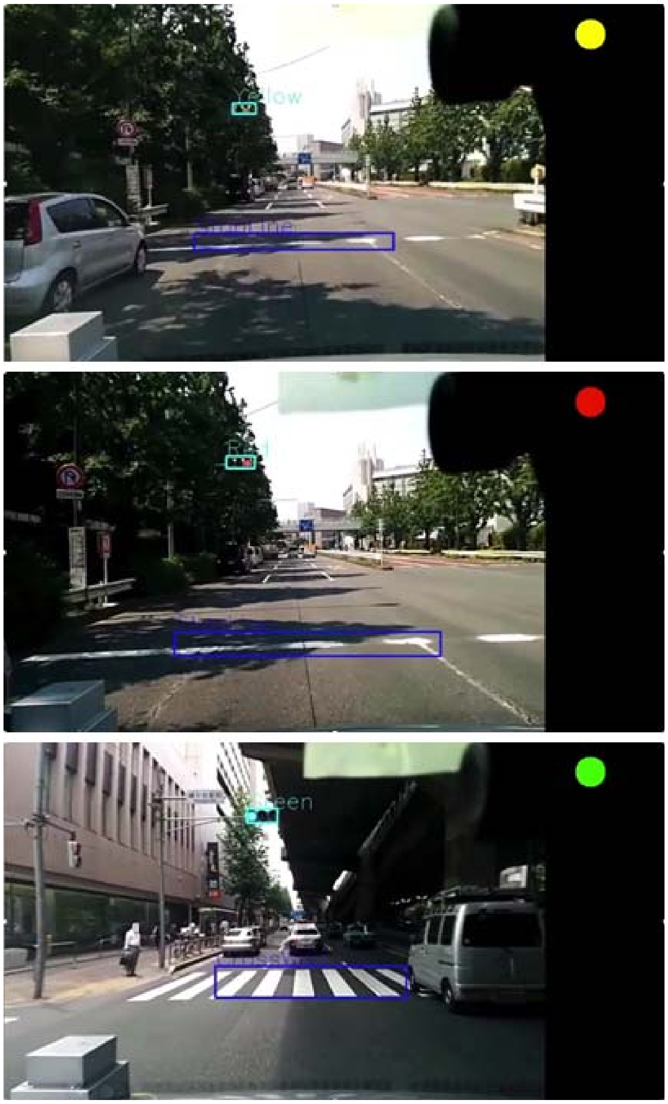


Fig. 9. Road facility detection results using the proposed network.

of the detection results from our method. Right part indicates the estimated traffic light state. In addition, the performance of the SSD is also indicated for the comparison in Table III. The evaluation result show that our proposed method has the almost similar detection accuracy as the SSD except for yellow traffic light. Our method performs much better than SSD for yellow light detection, because the targets in the similar classes, such as traffic lights, can share the features in the detection term of loss function and show the difference in the classification term. Besides of the detection precision, the processing time is also estimated. Our proposed detection network can achieve 2FPS in the smartphone, it is three times faster than the SSD network. This processing time proves that it is very possible to develop the intelligent DDR in the smartphone.

### C. Discussion for Intelligent DDR

In the proposed intelligent DDR system, the input data comes from the accelerometer, gyroscope and camera which are available in most of the current smartphones. The proposed intelligent DDR mainly consists of two parts: object detection in traffic scene and precise vehicle speed estimation. This paper uses the intersection scenario as an example to explain how the system works based on the road facility detection and speed estimation.

TABLE III
COMPARISON OF PRECISION AND PROCESSING TIME FOR ROAD FACILITY DETECTION

| | | SSD method | Our method |
|---|---|---|---|
| Detection precision | Stop line | 61% | 60% |
| | Crosswalk | 89% | 90% |
| | Green light | 81% | 81% |
| | Yellow light | 54% | 88% |
| | Red light | 90% | 90% |
| Processing time in smartphone | | 1500ms | 500ms |

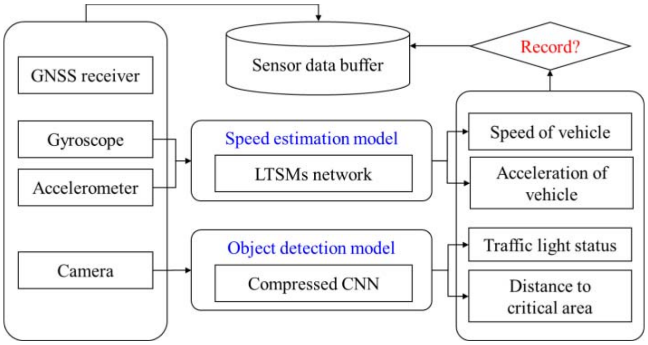


Fig. 10. Flowchart of intelligent DDR system.

The proposed DDR system can detect the traffic light, stop line and crosswalk. The traffic light status directly provide the information of the traffic situation. In addition, when the crosswalk and stop are detected, the distance from ego-vehicle to these critical areas can be estimated using inverse project mapping based on the gyroscope and accelerometer data [23], [24]. Moreover, the speed and acceleration of the vehicle can be provided by the speed estimation model and accelerometer, respectively. The proposed DDR system could provide four information: speed, acceleration, traffic light status and distance to these critical areas. That information can be logically organized based on the needs of the application, and can be integrated together for triggering the recording of DDR. For example, it is very important to record the driving behavior when the traffic light becomes yellow color. The proposed system can trigger the recording when the yellow traffic light is detected. In addition, the speed estimation function, stop line detection function can provide the more information to describe how the driver is approaching to the stop line when the traffic light is yellow.

This paper focuses on the speed estimation and road facility detection in the traffic scene. In the future, the vehicle and pedestrian detection will be included into the object detection model to make the DDR system have more intelligent functions.

## V. Conclusions and Future Work

This research presented a smartphone-based intelligent DDR system. Our proposed system consists of two parts: (1) a Long Short-term Memory (LSTM) neural network based model to estimate vehicle speed from data sequence of accelerometer and gyroscope. (2) A deep convolutional neural network based model to detect traffic lights, crosswalks, and stop lines. Comparing with conventional approaches for driving speed estimation, our proposed approach achieves more robust and precise prediction in the complicated urban areas. Considering the requirement of intelligent DDR and the limitation of smartphones' processing resources, road facility detection network is developed by using a lightweight architecture. On a smartphone platform (Google Pixel), our system could generate a high-accuracy detection at 2 FPS. In the future, the dynamic object detection will be included to provide the DDR system more functions.

## References

[1] C.-Y. Chan, "On the detection of vehicular crashes-system characteristics and architecture," *IEEE Trans. Veh. Technol.*, vol. 51, no. 1, pp. 180–193, Jan. 2002.

[2] P. L. Needham, "Collision prevention: The role of an accident data recorder (ADR)," in *Proc. Int. Conf. Adv. Driver Assistance Syst. (ADAS)*, Birmingham, U.K., Sep. 2001, pp. 48–51.

[3] M. Ayuso, M. Guillén, and A. M. Pérez-Marí, "Time and distance to first accident and driving patterns of young drivers with pay-as-you-drive insurance," *Accident Anal. Prevention*, vol. 73, pp. 125–131, Dec. 2014.

[4] *Drive Recorder Survey*. Accessed: Mar. 2015. [Online]. Available: http://www.jta.or.jp/kotsuanzen/pdf/H26drive_recorder_chosa.pdf

[5] *Mobile Fact Sheet*. Feb. 5, 2018. [Online]. Available: http://www.pewinternet.org/fact-sheet/mobile/

[6] C. Saiprasert, T. Pholprasit, and S. Thajchayapong, "Detection of driving events using sensory data on smartphone," *Int. J. Intell. Transp. Syst. Res.*, vol. 15, no. 1, pp. 17–28, Jan. 2017.

[7] I. Song, H.-J. Kim, and P. B. Jeon, "Deep learning for real-time robust facial expression recognition on a smartphone," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Las Vegas, NV, USA, Jan. 2014, pp. 564–567.

[8] M. Won, A. Mishra, and S. H. Son, "HybridBaro: Mining driving routes using barometer sensor of smartphone," *IEEE Sensors J.*, vol. 17, no. 19, pp. 6397–6408, Oct. 2017.

[9] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: Rich monitoring of road and traffic conditions using mobile smartphones," in *Proc. 6th ACM Conf. Embedded Netw. Sensor Syst. (SenSys)*, 2008, pp. 323–336.

[10] A. Allouch, A. Koubâa, T. Abbes, and A. Ammar, "Roadsense: Smartphone application to estimate road conditions using accelerometer and gyroscope," *IEEE Sensors J.*, vol. 17, no. 13, pp. 4231–4238, Jul. 2017.

[11] A. Thiagarajan *et al.*, "VTrack: Accurate, energy-aware road traffic delay estimation using mobile phones," in *Proc. 7th ACM Conf. Embedded Netw. Sensor Syst.*, 2009, pp. 85–98.

[12] D. A. Johnson and M. M. Trivedi, "Driving style recognition using a smartphone as a sensor platform," in *Proc. 14th IEEE Int. Conf. Intell. Transp. Syst. (ITSC)*, Washington, DC, USA, Oct. 2011, pp. 1609–1615.

[13] J. I. Meguro, Y. Kojima, N. Suzuki, and E. Teramoto, "Positioning technique based on vehicle trajectory using GPS raw data and low-cost IMU," *Int. J. Automot. Eng.*, vol. 3, no. 2, pp. 75–80, 2012.

[14] L.-T. Hsu, Y. Gu, and S. Kamijo, "Intelligent viaduct recognition and driving altitude determination using GPS data," *IEEE Trans. Intell. Veh.*, vol. 2, no. 3, pp. 175–184, Sep. 2017.

[15] M. Fazeen, B. Gozick, R. Dantu, M. Bhukhiya, and M. C. González, "Safe driving using mobile phones," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1462–1468, Sep. 2012.

[16] J. Yu *et al.*, "SenSpeed: Sensing driving conditions to estimate vehicle speed in urban environments," *IEEE Trans. Mobile Comput.*, vol. 15, no. 1, pp. 202–216, Jan. 2016.

[17] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, San Diego, CA, USA, vol. 1, Jun. 2005, pp. 886–893.

[18] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Kauai, HI, USA, Dec. 2001, p. 1.

[19] T. Watanabe, S. Ito, and K. Yokoi, "Co-occurrence histograms of oriented gradients for human detection," *IPSJ Trans. Comput. Vis. Appl.*, vol. 2, pp. 39–47, Jan. 2010.

[20] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, "Cascade object detection with deformable part models," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, San Francisco, CA, USA, Jun. 2010, pp. 2241–2248.

[21] Y.-T. Chiu, D.-Y. Chen, and J.-W. Hsieh, "Real-time traffic light detection on resource-limited mobile platform," in *Proc. IEEE Int. Conf. Consum. Electron.*, Taipei, Taiwan, May 2014, pp. 211–212.

[22] T. H.-P. Tran, C. C. Pham, T. P. Nguyen, T. T. Duong, and J. W. Jeon, "Real-time traffic light detection using color density," in *Proc. IEEE Int. Conf. Consum. Electron.-Asia (ICCE)*, Seoul, South Korea, Oct. 2016, pp. 1–4.

[23] Y. Gu, L.-T. Hsu, J. Bao, and S. Kamijo, "Integrating global navigation satellite system and road-marking detection for vehicle localization in urban traffic," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2595, pp. 59–67, 2016. [Online]. Available: https://trrjournalonline.trb.org/doi/abs/10.3141/2595-07

[24] Y. Gu, L.-T. Hsu, and S. Kamijo, "Passive sensor integration for vehicle self-localization in urban traffic environment," *Sensors*, vol. 15, no. 12, pp. 30199–30220, 2015.

[25] V. N. Murali and J. M. Coughlan, "Smartphone-based crosswalk detection and localization for visually impaired pedestrians," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, San Jose, CA, USA, Jul. 2013, pp. 1–7.

[26] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA, USA: MIT Press, 1998, pp. 255–258. [Online]. Available: https://dl.acm.org/citation.cfm?id=303704

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[28] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 1–9.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.

[30] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 142–158, Jan. 2016.

[31] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, Dec. 2015, pp. 1440–1448.

[32] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[33] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 779–788.

[34] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 6517–6525.

[35] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Computer Vision-ECCV*. Cham, Switzerland: Springer, 2016, pp. 21–37.

[36] Q. Wang, Y. Gu, J. Liu, and S. Kamijo, "DeepSpeedometer: Vehicle speed estimation from accelerometer and gyroscope using LSTM model," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Yokohama, Japan, Oct. 2017, pp. 1–6.

[37] Q. Wang, Y. Liu, J. Liu, Y. Gu, and S. Kamijo, "Critical areas detection and vehicle speed estimation system towards intersection-related driving behavior analysis," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Las Vegas, NV, USA, Jan. 2018, pp. 1–6.

[38] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[39] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.

[40] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 3156–3164.

[41] A. Graves, N. Jaitly, and A.-R. Mohamed, “Hybrid speech recognition with Deep Bidirectional LSTM,” in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Olomouc, Czech Republic, Dec. 2013, pp. 273–278.

[42] H. Sak, A. Senior, and F. Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” in *Proc. Interspeech*, 2014, pp. 338–342.

[43] M. Hermans and B. Schrauwen, “Training and analysing deep recurrent neural networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 190–198.

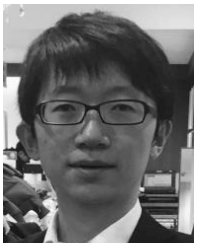
[44] M. D. Zeiler. (2012). “ADADELTA: An adaptive learning rate method.” [Online]. Available: https://arxiv.org/abs/1212.5701

[45] A. G. Howard *et al.* (2017). “MobileNets: Efficient convolutional neural networks for mobile vision applications.” [Online]. Available: https://arxiv.org/abs/1704.04861

[46] Y. Gu, L.-T. Hsu, and S. Kamijo, “Towards lane-level traffic monitoring in urban environment using precise probe vehicle data derived from three-dimensional map aided differential GNSS,” *IATSS Res.*, to be published. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0386111217300894

[47] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The PASCAL visual object classes (VOC) challenge,” *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.

**Yanlei Gu** (M’14) received the M.E. degree from the Harbin University of Science and Technology, China, in 2008, and the Ph.D. degree from Nagoya University, Japan, in 2012. He has been a Post-Doctoral Researcher at the Institute of Industrial Science, University of Tokyo, since 2013. His research interests include GNSS positioning, computer vision, and deep learning and their applications to ITS. He is a member of the IEEE ITS Society. He has served as the Organizing Committee Member for the IEEE ICVES2015 and the ITSC2017.



**Qianlong Wang** received the M.E. degree from The University of Tokyo, Japan, in 2018. His research interests include computer vision and deep learning and their applications to ITS.

**Shunsuke Kamijo** (M’97–SM’17) received the B.S. and M.S. degrees in physics, and the Ph.D. degree in information engineering from The University of Tokyo, Tokyo, Japan, in 1990, 1992, and 2001, respectively.

He was a Processor Design Engineer at Fujitsu Ltd., in 1992. From 2001 to 2002, he was an Assistant Professor, and has been Associate Professor since 2002. His research interests are computer vision, wireless communication, and their applications to ITS. His research focuses are autonomous vehicles, traffic video surveillance, traffic signal control, V2X communications, pedestrian and car navigations.

Prof. Kamijo is a member of the IEEE ITS Society, TRB, IEICE, and IATSS. He joined the International Program Committee of the ITS World Congress in 2011. He has been a member of the Board of Governors of the IEEE ITS Society since 2015 and an Executive Committee Member of the Society since 2017. He has served as the Vice-Chairman of the Program Committee for the ITS World Congress Tokyo 2013, the General Co-Chair for the IEEE ICVES2015 and the ITSC2017, and the International Program Chair for IEEE IV2017. He is an Editorial Board Member of the *International Journal on ITS research* (Springer) and *Multimedia Tools and Applications* (Springer).