室蘭工業大学
学術資源アーカイブ
Muroran Institute of Technology Academic Resources Archive

Human-Like Driving: Empirical Decision-Making System for Autonomous Vehicles

著者	LI Liangzhi, OTA Kaoru, DONG Mianxiong
journal or publication title	IEEE Transactions on Vehicular Technology
volume	67
number	8
page range	6814-6823
year	2018-04-03
URL	http://hdl.handle.net/10258/00009958

doi: info:doi/10.1109/TVT.2018.2822762

Human-Like Driving: Empirical Decision-Making System for Autonomous Vehicles

Liangzhi Li, *Student Member, IEEE,* Kaoru Ota, *Member, IEEE,* Mianxiong Dong, *Member, IEEE*

***Abstract*—The autonomous vehicle, as an emerging and rapidly growing field, has received extensive attention for its futuristic driving experiences. Although the fast developing depth sensors and machine learning methods have given a huge boost to self-driving research, existing autonomous driving vehicles do meet with several avoidable accidents during their road testings. The major cause is the misunderstanding between self-driving systems and human drivers. To solve this problem, we propose a human-like driving system in the paper to give autonomous vehicles the ability to make decisions like a human. In our method, a Convolutional Neural Network (CNN) model is used to detect, recognize and abstract the information in the input road scene, which is captured by the on-board sensors. And then a decision-making system calculates the specific commands to control the vehicles based on the abstractions. The biggest advantage of our work is that we implement a decision-making system which can well adapt to real-life road conditions, in which a massive number of human drivers exist. In addition, we build our perception system with only the depth information, rather than the unstable RGB data. The experimental results give a good demonstration of the efficiency and robustness of the proposed method.**

***Index Terms*—Self-driving, autonomous vehicles, collision avoidance system, vehicle control, machine learning.**

I. Introduction

In recent years, autonomous vehicles have become a popular topic, not only in the field of research but also in the application domain. Many encouraging approaches and prototypes have been made. However, the existing self-driving strategies focus too much on the "correctness", and, to some extent, overlook the human personality and social intelligence [1]. For example, during the road testing on February 14th, 2016, a Google autonomous car crashed into a municipal bus, which does not give way to the car but is predicted to slow or stop by the self-driving system. Google believes this accident is due to some misunderstanding and can be used as a valuable experience for the self-driving system [2]. They also state that their cars will have the knowledge that the larger vehicles are more unwilling to yield, and some software adjustments will be made to avoid this type of collision in the future.

In some complex conditions, autonomous vehicles should have the ability to perform *human-like* decisions and judgments, in which both correctness and social intelligence are essential. Undoubtedly, traffic regulations must be observed. But beyond that, predictions and understandings are also required. From determining whether the car behind is going to yield to avoiding a driver who seems drunk or tired, human drivers do these speculations all the time behind the wheel. Since human drivers will exist for a long time to come, self-driving systems with poor human understanding ability may struggle in the road testing, not to mention practical applications. Indeed, in most accident reports of self-driving vehicles, it is the human drivers that should bear the main responsibility. Therefore, to improve the self-driving performance in real-life road conditions, where most vehicles are in the hands of human drivers, we design an autonomous driving system in the paper to understand complicated road conditions and, based on that, make human-like decisions. As shown in Fig. 1, the proposed system prefers to thinking like a human, while traditional approaches make decisions according to the software settings.

In order to simplify the complexity of the decision-making system, we adopt a scene understanding subsystem to generate the abstractions from the raw input. This system only takes the 3D data from the light detection and ranging (LiDAR) sensor, which is able to generate stable and robust images in both light and dark environments. Using the past work experience, we propose a LiDAR-only scene understanding and abstraction method, as a component of the proposed system. In our method, RGB data is only used in the detection of nearby lane markings, which can always be covered by the headlight. And only depth information is needed in the detection and classification of other objects, e.g. cars, buses, trunks, etc.

In the paper, we focus on the autonomous decision-making system and implement it on the basis of a deep learning model. The main contributions of our work include:

- We work out an autonomous decision-making system to imitate human drivers' social intelligence, which can better adapt the self-driving vehicles to the real-life road conditions.
- We design an efficient training scheme for self-driving systems, which can significantly improve the quality and speed of data collection, and alleviate the tedious and time-consuming manual labeling process.
- We find an approach to analyze the reason that the deep learning system makes specific decisions. As far as we know, this is one of the first attempts in the self-driving area. In our opinion, this kind of analysis is of big significance to testing and validating autonomous driving systems.

Copyright (c) 2015 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

Authors are with the Department of Information and Electronic Engineering, Muroran Institute of Technology, Japan.
E-mail:{16096502, ota, mxdong}@mmm.muroran-it.ac.jp

Manuscript received xx xx, 20xx; revised xx xx, 20xx.

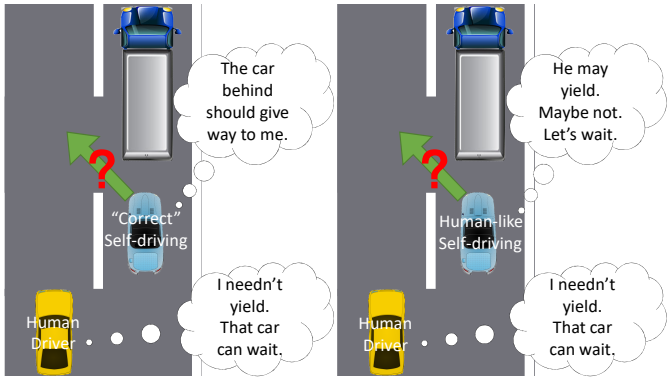


Fig. 1. Differences between traditional "correct" self-driving and the proposed human-like driving. The former acts on the "correctness" defined by the program, while the latter attempts to understand humanity and imitate their thinking.

II. Related Works

A. Scene Understanding

There have been several attempts to adopt scene understanding methods in the self-driving area [3]. Generally, scene understanding can be divided into three steps, i.e., image acquisition, object proposal and object recognition. We will give them a brief review.

Data acquisition for 2D images is simple. The key problem is how to obtain the 3D information efficiently. Fortunately, this is a fundamental task in robotic and self-driving systems, and many effective approaches have been proposed, e.g. [4]–[6].

Object proposal, which is also called the object detection and localization, is the first step in the whole scene understanding procedure. Only with properly proposed object candidates, the following object recognition can work correctly. While exhaustive search methods are able to correctly select objects with appropriately selected detection methods, it is too time-consuming and inefficient, considering the whole image space. Uijlings et al. [7] present a novel selective search method, which not only captures all scales but is fast to compute. The authors adopt a diverse set of complementary and hierarchical grouping strategies and work out an efficient, robust and stable proposal method. And then, Ren et al. [8] address this problem with fast-growing deep neural networks, and successfully achieve state-of-art performance. This work proves the feasibility and efficiency of the new emerging machine learning technologies when applied to object proposal problems. This is a totally new attempt and provides great insight into the object proposal field. Then some researchers begin to consider this problem in 3D images. Chen et al. [9] formulate this problem as minimizing a function involving free space, distance to the ground, point cloud densities and other information. Then the proposed method is able to place 3D box depending on the result to give out the object proposals.

There are also some scene recognition methods focused on the 3D images, such as [10]–[12]. We also work out a view-invariant 3D recognition method in [13], which can efficiently recognize the 3D scenes captured by the depth sensors.

Our main objective in this paper is to find a robust scene understanding and abstraction network for the subsequent driving decision-making process. Therefore, we combine the existing scene understanding approaches with the real-life road conditions and propose a perception system applicable to autonomous vehicles. The biggest difference between our work with existing methods is, we avoid the use of unstable RGB data for a more robust road scene understanding ability.

B. Driving Decision Making

With the rapid development of computer vision, engineering, networking, etc. [14]–[21], the newly-emerged autonomous vehicle has become a feasible and promising technology. However, the driving decision-making is still a research area under development. Although many researchers have proposed their theories and implementations [22]–[25], there are lots of problems to be resolved. Currently, most of the driving decision-making systems can be categorized into two major paradigms, i.e., behavior reflex approaches and abstraction calculation approaches.

Behavior reflex approaches directly map the input images to several pre-defined driving commands. LeCun et al. [26] propose an obstacle avoidance system for mobile robots. This system is trained from end to end to map raw input images to control commands. A CNN model is used in this system to learn the mapping relationship between images and driver's steering angles. And the mobile robot achieves an excellent performance in obstacle detection and path navigation. Hadsell et al. [27] work out a self-supervised learning method for mobile robots with long-range vision. In this system, a deep CNN model is trained to extract informative and meaningful features from captured images, and predict the transferability of the input scene. The classifier is real-time and can obtain obstacles and paths from 5 to 100 m.

Abstraction calculation approaches abstract the road condition to some representations for better understanding and calculation. Chen et al. [28] map the input scene to several key perception indicators which are directly related to the affordance of current road condition, and adopt a simple controller to control the self-driving vehicles. Simulation results demonstrate their system can generalize well to real driving images. However, their controller logic is oversimplified for real-life road conditions. Xiong et al. [29] combine deep reinforcement learning and safety-based control, and propose a self-driving and collision avoidance system, which can learn the driving policy in a stable and familiar environment. This is an interesting attempt, but their research is not complete and sufficient enough for a truly feasible autonomous driving system.

The proposed decision-making system in the paper belongs to the abstraction calculation approaches. And the most significant difference between our method and these existing approaches is, we put our emphasis on the human-like thinking ability. The proposed system has a complete and sophisticated control logic. Also, we combine the decision-making with the security control to make an efficient and secure self-driving system.

TABLE I
NOTATION LIST

Name	Meaning
$\mathcal{X}$	An input scene, which is a set of images
$\mathcal{A}$	An input abstraction set
$\mathcal{Y}$	Set of true labels of the vehicle category
$\mathcal{P}$	Set of true labels of the vehicle speed
$\mathcal{T}$	Set of true labels of the vehicle steering
$\mathcal{W}$	Weights of the deep model
$\mathcal{B}$	Biases of the deep model
L	The loss functions of the adopted deep models
λ	The pre-defined term weights for each loss functions
U	The repulsive potential
$\boldsymbol{F}$	The repulsive force
η	Positive scaling factor

III. SYSTEM OVERVIEW

A. Problem Definition and System Framework

Given a collection of road scene images $\mathcal{X}$, which are captured by on-board sensors during driving and consist all necessary information regarding the road condition at a certain moment, we can describe the goal of the proposed self-driving system as below. For each input scene $\mathcal{X}$, the perception subsystem will recognize as many as possible vehicles on the road and give them accurate labels. Then the decision-making subsystem should give out some advice $\mathcal{D}_h$ regarding the speed and steering, according to the abstraction $\mathcal{A}$ of current road condition, and, ultimately, generate the precise driving command d_{fin}, considering the security policy. We list the adopted notations in Table I.

We introduce the system framework in Fig. 2 to give a brief description of the proposed system. As can be seen, the vehicles use the CNN models to respectively perform the detection and recognition task, then the abstraction results will be transferred to the decision-making system. As for the markings, they are processed by several simple edge detection methods. For brevity, they are omitted in the paper.

After the perception process, the decision-making network calculates the driving decisions based on these abstractions. The decision-making model is the focus of this research, and we give it a detail introduction in Section IV. In addition, an influence analysis module is included in the model to clarify the specific relationship between the road conditions and the output decision.

B. 3D Perception and Abstraction

One major characteristic of the abstraction calculation approaches is that the abstraction of the road condition is generated before making driving decisions. In this procedure, a robust and efficient 3D perception method is essential to obtain the accurate understanding of the input images, which is captured by the sensors equipped on the self-driving vehicles. This task is also called scene understanding, one hot topic in computer vision area. However, as mentioned above, the traditional methods struggle in the perception and abstraction of road images, due to their dependences on unstable RGB data. We propose a specially-designed perception method to understand the input scenes, in order to improve the performance

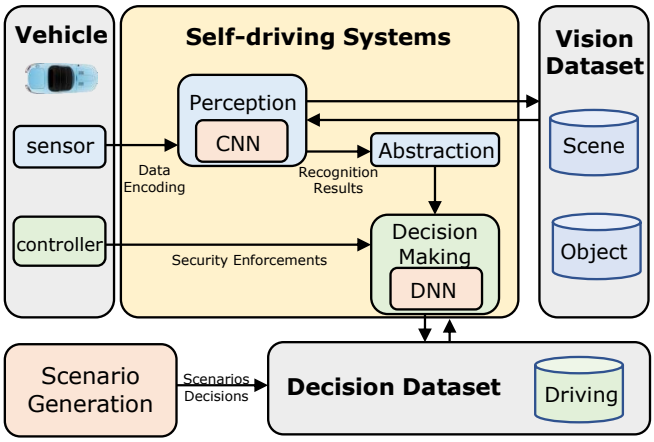


Fig. 2. System framework.

(a) (b)

Fig. 3. The perception and abstraction of road condition. (a) Road simulation using the open racing car simulator (TORCS). (b) Generated abstraction using the proposed road perception method.

of vehicle perception, and work out an abstraction approach for simplified road representation. The road data comes from two sources: the famous vision dataset for autonomous driving KITTI [30] and the open racing car simulator (TORCS). KITTI provides a great number of real-world road scenes, which is captured by a real car equipped with a laser scanner and a GPS localization system. And TORCS is a fully customizable car racing simulation environment, where lots of cars and road types are available. Both KITTI and TORCS are adopted in the benchmarking and simulation tests.

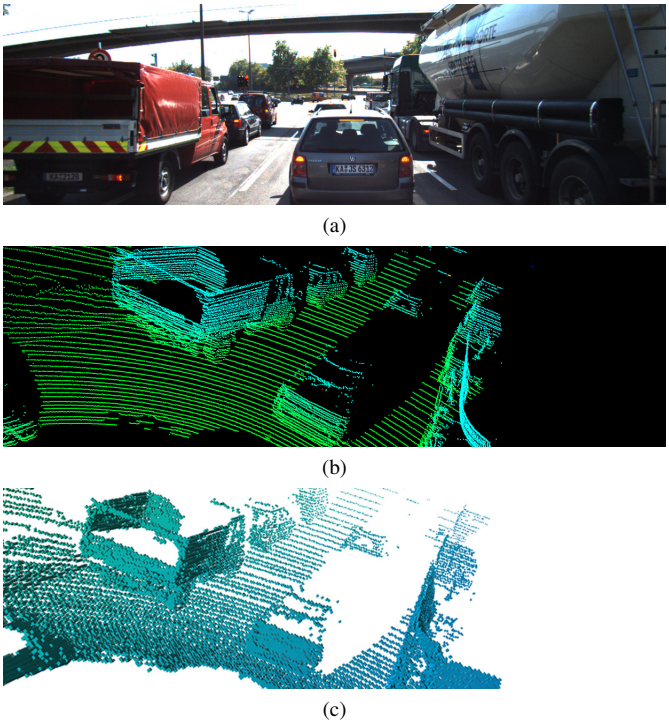
In short, the main purpose of the proposed perception and abstraction method is to understand the captured road images and generate a simplified representation for the following decision-making process. As shown in Fig. 3, the ego car in TORCS is equipped with depth sensors and can send the depth images to the scene understanding network in real time, and then the current road condition is abstracted by the proposed scene understanding method for the planar representation, which can be directly imported into the decision-making model. The perception and abstraction method is detailed below.

We propose the road condition understanding network (RCUN) to understand the road scenes, and adopt the famous Alex model [31] as the recognition network, and its modified version, which is compatible with 3D input data, as the vehicle proposal network. The major difference between RCUN and other perception methods is that we utilize different data en-

codings in these two models. The original raw data generated by the depth sensors is in the point cloud format, which is shown in Fig. 4(b). There are two ways to process these data. The first one is to map these data into RGB space, using the Point Cloud Library (PCL) [32], and obtain the 2D colored representation. The other one is the 3D representation, in which the point cloud data is transformed into volumetric encoding, which could be imported into computational models with 3D spatial information preserved, using Octomap [33], as shown in Fig. 4(c). The reason for this design is based on the following facts.

These two approaches have their advantages and disadvantages. One huge superiority of 2D representation is that there have been extensive research works regarding 2D image descriptors and feature engineering, which is of great help in our designing of 2D-encoded scene understanding models. And also, tremendous 2D image datasets have been published, and even more well-trained models relying on these datasets have been made available, on whose basis we can establish our own models. According to [34], low-layer features are not specific to one particular task or dataset and can be applied to many other networks. Similar to the physiological visual mechanism of the human, neurons of lower layers merely deal with the most specific and simplest tasks, which are usually homogeneous between many different applications. Therefore, the existing models are adopted as the basis of our own model with further fine-tuning, learning abstract features in high-layers once again and improving the training speed and network precision of the newly-designed model. On the contrary, works on 3D images are yet to be developed and improved. Few large-scale 3D image databases or effective trained models have been proposed. Another advantage of 2D representation is its high resolution. Due to some restrictions in principle and implementation, the volumetric representation must be encoded in a lower resolution (0.02 m in Fig. 4(c)), whereas 2D representations can preserve image details well. On the other hand, the volumetric representation also has some advantages. Unlike 2D encoding, 3D representation can indicate the 3D spatial information of the original scene, which is highly valuable for some applications. Therefore, we design our system with two data encoding types, which are respectively for the 3D detection and recognition.

After training, these two networks are able to perform the proposal and recognition tasks.



(a) (b) (c)

Fig. 4. Road scene encoding and representation [30]. (a) The original scene captured by the RGB camera. (b) The point cloud captured by the depth sensor. (c) 3D encoding.

IV. Decision Making

After obtaining the abstraction of road conditions, the system can work on the corresponding driving decisions. Compared to behavior reflex approaches, which directly map the input images to several pre-defined driving commands, the proposed method, using the abstractions instead of original images to calculate reactions, is much simplified in the inherent logic and structure, and as a result, is much more trainable and can achieve higher network precisions.

The generated abstraction is shown in Fig. 3(b), however, the abstraction data is still a bit complicated to be inputted into the decision-making network, due to the massive possibilities of relative vehicle positions. For further simplification, we define lots of grids all over the necessary road area, and then each vehicle can be subjected to one grid, which can be represented by one numerical value. This method gives a huge convenience for data encoding and network input because only a fixed number of girds exist in the interested zone of the road. We give these grids two attributes to reflect the type and speed of the vehicle on the road and adopt the girds as the input matrix of the proposed network. More precisely, each grid has two attributes, including one discrete attribute value, representing the specific vehicle type including sedan, bus, trunk, ego vehicle, none, unobservable, etc., and one continuous value, representing the vehicle current speed. If there is no vehicle in a grid, then its speed attribute is set to zero. As shown in Fig. 7(a), the blue car is the ego vehicle, and all the grids are labeled with a specific vehicle type and corresponding vehicle speed. Notably, although the on-board sensors improve a lot in the visual performance nowadays, it is not possible to obtain accurate road conditions in all grids. Especially, the vehicle perception system may fail to output an exact result, due to the occlusion. In this situation, the unknown or indefinite grids are labeled with "unobservable" and painted blue in the figure.

In this section, we will detail the proposed decision-making system, and also introduce an analysis method to clarify the relationship between the road factors and the final decision.

A. Decision Model

We propose a six-layer decision-making network (DMN) to learn human decision-making behaviors, as shown in Fig 5.

The input is the generated abstractions of road conditions. And the main objective is to make driving decisions based on these input data. The proposed model contains five hidden layers and one output layer. There are 1028 neurons in each of the first two layers, and 512 in each of the following three layers. This is a simple design because we focus on the data engineering and network learning process. We believe, using an effective training scheme, the proposed method can extract sufficient information from the well-processed input data, even with a simple model. Due to the superior feature extraction ability of deep model, we demonstrate that six layers are enough to obtain the relationship between road condition and driving decisions.

The distinctiveness is that we design the DMN as a two-part structure. Two individual sub-networks are adopted to respectively calculate the speed and steering commands. Since the low-level features are usually similar in neural networks, the first three fully-connected layers are shared between two sub-networks for training efficiency. There are two more layers in each sub-network to extract the high-level features, which is highly correlated to the target of each sub-network.

There are n input abstractions $\mathcal{A} = \{a_1, a_2, ..., a_n\}$ in DMN, with their corresponding labels a_i, including the speed labels $\mathcal{P} = \{p_{a_1}, p_{a_2}, ..., p_{a_n}\}$ and steering labels $\mathcal{T} = \{t_{a_1}, t_{a_2}, ..., t_{a_n}\}$. $t \in \{t_1^*, t_2^*, t_e^*\}$ are the possible labels, representing left and right lane changing or keeping the current lane, and p should be real values representing the possible vehicle speed. $\boldsymbol{h_{spd}(a_i)}$ and $\boldsymbol{h_{str}(a_i)}$ are the results of each layer.

According to the multi-task loss introduced in [35] and [36], the loss function is defined as follows.

$$L = \lambda_1 L_{\text{spd}} + \lambda_2 L_{\text{str}} \tag{1}$$

where the loss functions, including L_{spd} and L_{str}, are adopted for different goals, and their parameters, including λ_1 and λ_2, are pre-defined to give them respective weights according to their importance.

L_{spd} is utilized to calculate the acceleration or deceleration commands and it is a regression of desired vehicle speed. Using the smooth L_1 loss [36], L_{spd} can be defined as

$$L_{\text{spd}} = \frac{1}{n}\sum_{i=1}^{n} \text{smooth}_{L_1}(\boldsymbol{h_{spd}(a_i)} - \boldsymbol{p_{a_i}}) \tag{2}$$

L_{str} is calculated using the softmax function. It is adopted to compute the steering commands, i.e. changing lanes or not. It can be expressed as

$$L_{\text{str}} = -\frac{1}{n}\left[\sum_{i=1}^{n}\sum_{j=1}^{3} \mathbf{1}\{y_{a_i} = y_j^*\} \log \frac{e^{h_{\text{str}}^{(L)}(a_i)}}{\sum_{\phi=1}^{3} e^{h_{\text{str}}^{(\phi)}(a_i)}}\right] \tag{3}$$

where $h_{\text{str}}^{(j)}(a_i)$ varies from 0 to 1, according to the momentum to change to different lanes.

Also, the SGD strategy is adopted in this work to perform network training and the output results of the proposed network are regarded as driving advice $\mathcal{D}_h$ for the final decision-making strategy, which is detailed in the next section.

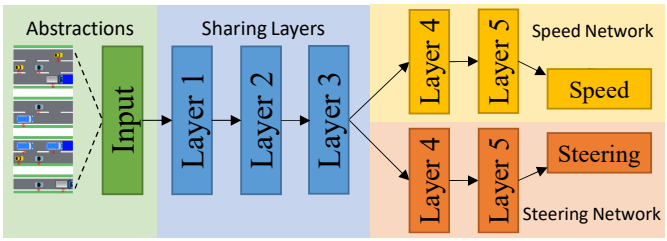


Fig. 5. Architecture of the decision-making network.

Algorithm 1: Decision-making Logic

Input: Current Road Condition c
Output: Driving Decisions $\boldsymbol{d_{fin}}$

```
/* Make decisions constantly.                    */
while IsDriving do
    A = GenerateAbstractions(c)
    D_h = ComputeAdvices(A)
    D_s = CollisionAvoidance(A)
    /* Examine D_h in confidence order           */
    for d_h^spd, d_h^str ∈ D_h do
        d_fin^spd = d_h^spd + d_s^spd,  d_fin^str = d_h^str + d_s^str
        d_fin = (d_fin^spd, d_fin^str)
        d_fin = RegulationsChecking(d_fin)
        /* Final Security Check.                 */
        if isSafeDriving(d_fin) = True then
            output(d_fin)
            break
    KeepSafeDriving()
```

B. Decision-making Strategy

The whole process of the proposed decision-making system is presented in Algorithm 1. As mentioned above, at first, the RCUN model generates abstractions of the inputted images regarding current road conditions. And then DMN model can calculate and output the human-like decisions. Although these decisions are already very reasonable, they are still not yet fully adaptable for actual driving situations, due to the security reasons. We combine the driving advice generated by DMN with a safety enforcement method, namely the repulsive potential field (RPF) [29]. RPF is a famous method used for robotic path planning. Its main objective is to control the robots to safely reach destinations while avoiding collisions. RPF mainly relies on the repulsive force of possible obstacles, which is very suitable for driving safety control. In detail, we regard all other vehicles and both road edges as the obstacles and then calculate their repulsive forces. The repulsive potential formula can be expressed as

$$U_{\text{rep}}(\boldsymbol{o}) = \begin{cases} \frac{1}{2}\eta(\frac{1}{d(\boldsymbol{o})} - \frac{1}{d}), & d(\boldsymbol{o}) \le d \\ 0 \quad , & d(\boldsymbol{o}) > d \end{cases} \tag{4}$$

where $\boldsymbol{o}$ represents the obstacle object, $d(\boldsymbol{o})$ is the distance between the obstacle and ego vehicle, η is a positive scaling factor. And, d is a pre-defined constant value representing

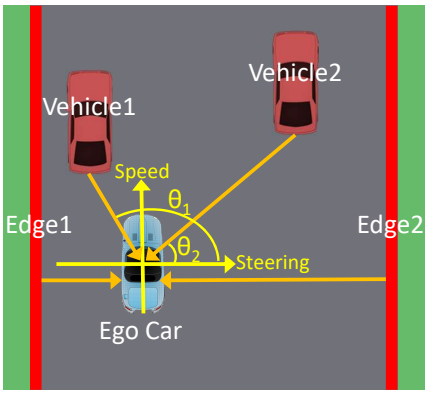


Fig. 6. Repulsive force and security enforcement.

for the maximum obstacle distance. Any obstacles beyond the distance d are neglected. Having the repulsive potential formula, the repulsive force can be expressed as

$$\begin{aligned} \boldsymbol{F_{rep}}(\boldsymbol{o}) &= -\nabla U_{\text{rep}}(\boldsymbol{o}) \\ &= \eta(\frac{1}{d(\boldsymbol{o})} - \frac{1}{d})(\frac{1}{d^2(\boldsymbol{o})})\nabla d(\boldsymbol{o}) \\ &= \eta(\frac{1}{d(\boldsymbol{o})} - \frac{1}{d})(\frac{1}{d^2(\boldsymbol{o})})\frac{\boldsymbol{o}-\boldsymbol{e}}{\parallel \boldsymbol{o}-\boldsymbol{e} \parallel} \end{aligned} \tag{5}$$

As shown in Fig 6, the objects in red color are regarded as the obstacles, including two edges and two vehicles in front of the ego car. The orange lines represent the repulsive force, and the yellow coordinate represents the driving decisions. It can be seen that the repulsive force and the ego car form an angle θ, and the forces are respectively exerted on the steering and speed axis. Then, the RPF security enforcement can be written as

$$d_s^{spd} = -\sum_{\boldsymbol{o}}^{\text{objects}} \parallel \boldsymbol{F_{rep}}(\boldsymbol{o}) \parallel sin(\theta_{\boldsymbol{o}}) \tag{6}$$

$$d_s^{str} = -\sum_{\boldsymbol{o}}^{\text{objects}} \parallel \boldsymbol{F_{rep}}(\boldsymbol{o}) \parallel cos(\theta_{\boldsymbol{o}}) \tag{7}$$

In addition, since the steering command is a discrete value, several thresholds are set to discretize d_s^{str}.

We design the final decision-making equation as below.

$$\begin{aligned} \boldsymbol{d_{fin}} &= \omega_1 \boldsymbol{d_h} + \omega_2 \boldsymbol{d_s} \\ s.t. \quad & \omega_1 + \omega_2 = 1 \end{aligned} \tag{8}$$

where $\boldsymbol{d_{fin}}$ is the calculated final decision, $\boldsymbol{d_h}$ and $\boldsymbol{d_s}$ represent the human-like driving advice and safety enforcement, respectively, and ω_1, ω_2 are the custom weights. In addition,

$$\begin{aligned} \boldsymbol{d_h} &= [d_h^{spd}, d_h^{str}] \\ \boldsymbol{d_s} &= [d_s^{spd}, d_s^{str}] \end{aligned} \tag{9}$$

are the output of DMN and RPF. As shown in Algorithm 1, after the final decision is generated, it is regularized to follow the traffic regulations, for example, driving along the lane, and examined with some final security check, which is defined according to local situations. Ultimately, the system can give out a safe and human-like driving decision.

Algorithm 2: Scenarios Generation

Input: Desired Scenarios Number n; Vehicle Type (including "none") $\mathcal{V}$ and their Propotions $\mathcal{P}_{\mathcal{V}}$; Vehicle Type $\mathcal{R}$, their Propotions $\mathcal{P}_{\mathcal{R}}$ and Speed Distribution in each road type $\mathcal{D}_{\mathcal{S}}$
Output: Simulation Scenarios $\mathcal{S}$

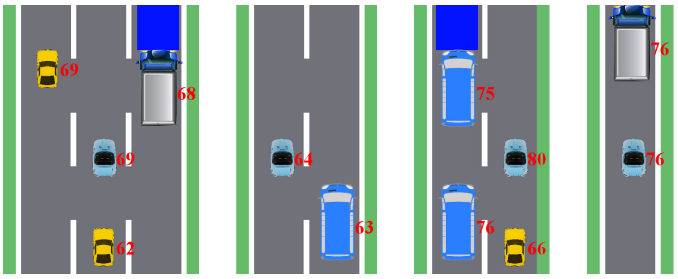
```
scenario_list = null
while length(scenario_list) < n do
    /* Generate a road r.                          */
    r = RandomRoad(R, P_R)
    /* Try all available positions.                */
    for grid ∈ r do
        /* Generate a new vehicle v.               */
        v = RandomVehicle(V, P_V)
        if ExistCarInFront(v) is False then
            /* Assign vehicle speed v.             */
            v.speed = RandomSpeed(v, r, D_S)
        else
            /* Adjust v according to the
               speed of vehicles in front.         */
            v.speed = SpeedAdjust(r)
        r.append([v,grid])
    scenario_list.append(r)
return scenario_list
```

C. Network Training Scheme

As the proposed decision-making system is based on a machine learning model, an important problem is how to obtain sufficient learning materials for the underlying neural network. One traditional way is, obviously, to record actual human driving behaviors, using either a real car or a custom gaming simulation environment. However, it is very inefficient to conduct such data acquisition task, which usually needs several months to get adequate road scenarios and driving decisions, even for a small-scale dataset.

We implement a novel generation strategy of training material to efficiently create driving scenarios for human reaction recording, as shown in Algorithm 2. At first, the road type is randomly selected from several templates according to the pre-defined occurrence possibilities. Then the vehicles are also generated in a similar way and get assigned to random grids of the newly generated road. Each vehicle will be set with a specific speed value, and be adjusted to be as realistic as possible. The generated scenarios are shown in Fig. 7. The type and relative position of the vehicles are all randomly selected, on the basis of some common situations, traffic rules, driving habits, etc. The last step is to import these scenarios into some driving environments, such as the TORCS, and record the driving decisions of test drivers. To improve the sense of reality and interactivity, some peripheral equipment can be used to emulate the actual driving experience, e.g. steering wheel and pedals. Before each scenario, a brief buffer time is available for the test drivers to get familiar with the current road condition.

Compared to some commonly used data recording methods, the proposed scheme has a significant advantage at the



(a) (b) (c) (d)

Fig. 7. Generated driving scenarios for network training. Subfigure (a)(b)(c)(d) represent four different road types, and the blue car represents the ego vehicle.

efficiency, and more importantly, the representativeness of the generated scenarios. In fact, there are a large number of unnecessary testing scenarios in the continuous driving process. Therefore, our scheme can avoid this problem by generating discontinuous scenarios.

Algorithm 3: Influence Analysis

Input: Trained Network N, Specific Road Abstraction $\mathcal{A}$
Output: Visualized Unit Influences

```
/* Check all the unit values in both
two input channels.                          */
for att ∈ [vehicle_type, vehicle_speed] do
    influences_vector = null;
    original_output = h^N(A)
    for u ∈ A_att do
        /* Modify the unit value.           */
        if att is vehicle_type then
            A_new = ChangeType(A, u)
        else
            A_new = ChangeSpeed(A, u)
        new_output = h^N(A_new)
        /* Compute the impact of the
        changed unit value.                  */
        influence = new_output − original_output
        influence = Normalize(influence)
        influences_vector ← influence
    /* Generate the visualization.          */
    Visualize(influences_vector)
```

D. Influence Analysis

We find it essential to figure out why the proposed neural network can make human-like reactions to various road conditions. Which parts of the input abstractions play a key role in the final decision? What is the network focused on among all the input values including the vehicle types and speed in every road grid? Indeed, it is not easy to accurately visualize the specific influences of the input units in a neural network. In this section, we implement an ingeniously-designed visualization method to analyze the mechanisms behind the proposed decision-making network. There are two major benefits of this analysis method. The first one is that, with an intuitive heatmap, self-driving researchers can easily find the specific areas and information which the decision-making system is really interested in. And as a result, researchers can find ways to improve the accuracy or clearness of these valuable data, e.g., increasing the resolution of the back depth sensor, refining the vehicle category of the road perception system, improving the accuracy of velometer, etc. The other benefit is, the generated heatmap is an important criterion reflecting whether the decision-making network is overfitting or not. Since the network architecture is defined according to subjective experience, it is possible that the deep network is designed with too many layers or neurons. In this situation, if no enough training material is available, the network will suffer the overfitting problem. Although an overfitted network performs well from the perception of training loss, it may easily fail in some other road conditions. It is very difficult to judge the overfitted networks using simple precision tests. However, the overfitted networks show inexplicable and meaningless heatmaps in most cases. By examining the generated influence maps carefully, researchers can find some symptoms of the overfitting problem. This algorithm is introduced below.

In a trained network, the influence of one input u, which is a single value in the input vector, to the output of layer θ is presented below

$$\boldsymbol{h}_{\boldsymbol{\theta}}^{\boldsymbol{N}}(u) = \begin{cases} \boldsymbol{W}_{\boldsymbol{\theta}}\boldsymbol{h}_{\boldsymbol{\theta}}(u) + \boldsymbol{b}_{\boldsymbol{\theta}}, & \theta \geq 2 \\ \boldsymbol{W}_{\boldsymbol{\theta}}u + \boldsymbol{b}_{\boldsymbol{\theta}} \quad , & \theta = 1 \end{cases} \tag{10}$$

Notice that, with a fixed $\boldsymbol{W}_{\boldsymbol{\theta}}$ and $\boldsymbol{b}_{\boldsymbol{\theta}}$, the output of every layer in the neural network, including the final output, is only related to the input u. Therefore, the contribution of each input values can be inferred using Algorithm 3. In the beginning, each vector unit of both two attributes, i.e. grid type and vehicle speed, will be checked for the influence calculation. Its value is modified with a minor adjustment, and the difference between the original and new output results is calculated to reflect the contribution of this grid. The numerical results are normalized and transformed into heatmaps.

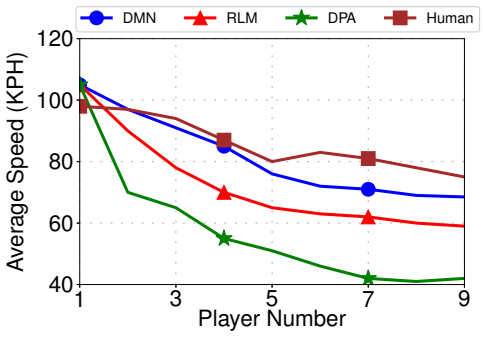
An experiment is conducted in Section V-B to demonstrate this analysis method.

V. Performance Evaluation

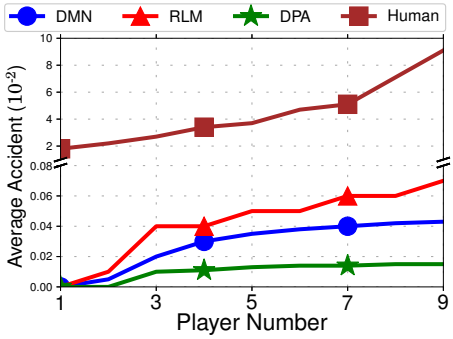
Several experiments are conducted in this section to evaluate the performance of the proposed self-driving system. First, we carry out a driving simulation to test the abilities of decision-making method. Then a visualization of unit influences is presented, using the strategy described in Section IV-D.

A. Decision-making System Evaluation

The best way to evaluate the performance of driving systems is to conduct the road tests in a simulation environment. As mentioned above, TORCS is a common choice for the testing platform in driving simulations. Therefore, we also adopt TORCS in this experiment for better comparability. In this section, the focus problem is the rationality and security of generated driving decisions, while the scene understanding ability is not the emphasis. Therefore, the exact abstractions



(a)



(b)

Fig. 8. Performance evaluation in driving simulation. (a) Comparison results of average driving speed. (b) Comparison results of average driving accidents

of the road conditions are directly passed to the self-driving methods by the TORCS platform, rather than the scene understanding methods. Then, the key problem is the experiment design, which should give a comprehensive test on both the advance speed and driving security.

The testing details are introduced below. First, we select and modify an existing roadmap in TORCS, which is a two-lane street with a total length of $15,000$ meters, making clearer lane markers and richer roadside views. Then, an autonomous vehicle, controlled by different self-driving methods, is added to the track as player Ego. Up to eight human players are also included in the same track, so the self-driving vehicle must take other players into account when driving on the road. There are nine test settings with different human numbers for one self-driving method, and ten tests are conducted for each test setting. The vehicle speed and total accident number are recorded during these tests. Fig. 8 presents the average results of vehicle speed and accident number. Four driving methods are adopted for player Ego, i.e., the proposed DMN method, the reinforcement learning method (RLM) [29], the direct perception approach (DPA) [28], and manual driving (Human), which is set for better understanding.

It can be seen that, in Fig. 8(a), with the increase of player number, all the testing vehicles show slower average speed. Among the self-driving methods, the proposed DMN shows an obvious advantage beyond other two methods, due to its intelligent decision-making ability. The RLM method is implemented by us according to the principle introduced in the original paper. It is a novel design, but does not consider the human driving behaviors and social intelligence. Therefore, it is slightly inferior in this test with human participants. DPA is focused on the driving perception, and only has a simple driving logic, which mainly cares about the security rules. So it does not perform well in the speed testing. Unexpectedly, the increase of player number has little influence on the human driver, possibly because the human drivers have the social intelligence ability and prefer to aggressive driving strategies. Fig. 8(b) presents the results of security testing, i.e., the average accident number. Contrary to the speed testing results, DPA outperforms other methods because the security regulations are carried out rigorously in DPA. And our DMN has the second best security performance, due to the safety enforcement and, more importantly, the human-like decision-making ability. RLM also considers the safety requirement, but is outperformed by DMN due to the lack of social intelligence. Because the road conditions are directly sent to self-driving systems, they can easily beat the human driver in this testing, which, of course, is different with the real-life driving environment, where the road conditions must be understood and abstracted by the perception system. However, as a performance evaluation, these results can sufficiently demonstrate the feasibility and stability of our decision-making method.

B. Influence Analysis and Visualization

As it has been demonstrated that our self-driving system is able to perform sound and safe decisions, an additional experiment is conducted to clarify the relationship between the road conditions and the driving decision. Fig. 9 shows the visualization results of two randomly generated scenarios. The input abstractions are presented in the first column of Scene A and B in Fig. 9, and the second and third column give out the unit contributions to the speed decision network and steering decision network. The output decision of Scene A is to keep the current lane and pick up speed; the decision of Scene B is to deceleration and move away from the front truck. It can be seen that, in Scene A, DMN notices there is an extensive free area in front of ego-car. DMN also judges that the cars in neighbor lanes are unlikely to change lanes, and, therefore, have little impact on the final decision. In Scene B, DMN decides not to change lane because of the unknown area in the top-right corner, which is reflected in the vehicle-type channel in the steering network. Instead, DMN prefers to slow down and keep the speed slightly slower than the front truck.

Interestingly, the visualized unit contributions give us deep insights into the mechanism of the decision-making neural network. We find several interesting results after the unit contributions are transformed into heatmaps. These results are reasonable and explainable. The way DMN considers its decisions is so similar to the human mind that, we believe, it can achieve more amazing performance with a larger dataset. The experiment results demonstrate that the proposed method does have the ability to perform human-like decisions and adapt well to various road conditions.

VI. Conclusion

A human-like autonomous driving system is proposed in the paper. This system mainly includes two parts, i.e. a road scene

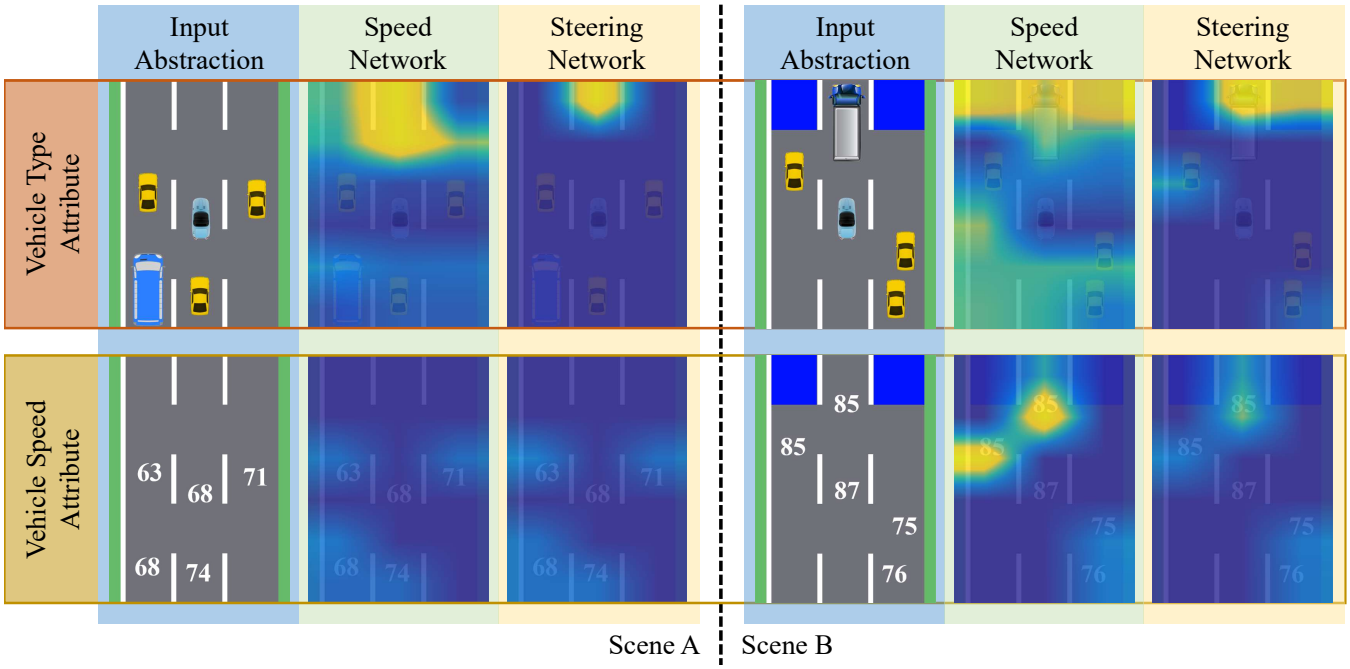


Fig. 9. The visualization results of unit influence. Scene A and B are two random generated scenarios. The first column is the abstracted road condition; the second and third column respectively show the contributions of each input unit to the speed and steering network.

perception method and an empirical decision-making network. One obvious difference with the existing approaches is that it can imitate human drivers' social intelligence, which can better adapt the self-driving vehicles to the real-life road conditions. In addition, we implement an efficient training scheme to improve the quality and speed of data collection, and alleviate the tedious and time-consuming manual labeling process. We also find a feasible approach to analyze the possible influence factors in the decision-making process, which can help in the testing and validating of autonomous driving systems. The experimental results prove that the proposed method is efficient, and shows meaningful analysis results in the visualizations of unit influence.

Future work includes developing a more efficient optimization method to decrease the time cost of the training process. Also, some information may be extracted from RGB data, even when it is incomplete and unstable, which can be a useful supplement to our method. In addition, we will extend our work by conducting more driving testings in order to further expand the dataset for network training.

ACKNOWLEDGMENT

This work is partially supported by JSPS KAKENHI Grant Number JP16K00117, JP15K15976, and KDDI Foundation. Mianxiong Dong is the corresponding author.

REFERENCES

[1] T. B. Sheridan, "Human–robot interaction status and challenges," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 58, no. 4, pp. 525–532, 2016.

[2] Q. Yuan, Y. Gao, and Y. Li, "Suppose future traffic accidents based on development of self-driving vehicles," in *Man-Machine-Environment System Engineering*. Springer, 2016, pp. 253–261.

[3] T. Chen and S. Lu, "Accurate and efficient traffic sign detection using discriminative adaboost and support vector regression," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 4006–4015, June 2016.

[4] C. Hne, L. Heng, G. H. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys, "3d visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection," *Image and Vision Computing*, vol. 68, no. Supplement C, pp. 14 – 27, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0262885617301117

[5] V. De Silva, J. Roche, and A. Kondoz, "Fusion of lidar and camera sensor data for environment sensing in driverless vehicles," *arXiv preprint arXiv:1710.06230*, 2017.

[6] X. Fan, L. Zhang, B. Brown, and S. Rusinkiewicz, "Automated view and path planning for scalable multi-object 3d scanning," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 239:1–239:13, Nov. 2016. [Online]. Available: http://doi.acm.org/10.1145/2980179.2980225

[7] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[9] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals for accurate object class detection," in *Advances in Neural Information Processing Systems*, 2015, pp. 424–432.

[10] L. A. Alexandre, *3D Object Recognition Using Convolutional Neural Networks with Transfer Learning Between Input Channels*. Cham: Springer International Publishing, 2016, pp. 889–898.

[11] Y. Xiang, W. Kim, W. Chen, J. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, and S. Savarese, *ObjectNet3D: A Large Scale Database for 3D Object Recognition*. Cham: Springer International Publishing, 2016, pp. 160–176.

[12] N. Sedaghat, M. Zolfaghari, and T. Brox, "Orientation-boosted voxel nets for 3d object recognition," *arXiv preprint arXiv:1604.03351*, 2016.

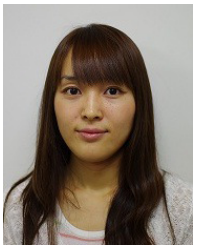
[13] L. Li, K. Ota, M. Dong, and W. Borjigin, "Eyes in the dark: Distributed scene understanding for disaster management," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 12, pp. 3458–3471, Dec 2017.

[14] C. Chen, S. Zhu, X. Guan, and X. S. Shen, *Wireless sensor networks: Distributed consensus estimation*. Springer, 2014.

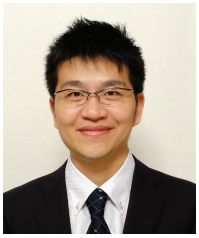
[15] C. Chen, T. H. Luan, X. Guan, N. Lu, and Y. Liu, "Connected vehicular transportation: Data analytics and traffic-dependent networking," *IEEE Vehicular Technology Magazine*, vol. 12, no. 3, pp. 42–54, Sept 2017.

[16] Z. Su, Y. Hui, T. H. Luan, and S. Guo, "Engineering a game theoretic access for urban vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 4602–4615, June 2017.
[17] Z. Su, Q. Xu, Y. Hui, M. Wen, and S. Guo, "A game theoretic approach to parked vehicle assisted content delivery in vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 7, pp. 6461–6474, July 2017.
[18] H. Li, M. Dong, and K. Ota, "Control plane optimization in software-defined vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 10, pp. 7895–7904, Oct 2016.
[19] H. Li, K. Ota, and M. Dong, "Network virtualization optimization in software defined vehicular ad-hoc networks," in *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, Sept 2016, pp. 1–5.
[20] K. Ota, M. Dong, H. Zhu, S. Chang, and X. Shen, "Traffic information prediction in urban vehicular networks: A correlation based approach," in *2011 IEEE Wireless Communications and Networking Conference*, March 2011, pp. 1021–1025.
[21] K. Ota, M. Dong, J. Gui, and A. Liu, "Quoin: Incentive mechanisms for crowd sensing networks," *IEEE Network*, vol. PP, no. 99, pp. 1–6, 2018, doi: 10.1109/MNET.2017.1500151.
[22] M. M. Atia, S. Liu, H. Nematallah, T. B. Karamat, and A. Noureldin, "Integrated indoor navigation system for ground vehicles with automatic 3-d alignment and position initialization," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1279–1292, April 2015.
[23] J. Ji, A. Khajepour, W. Melek, and Y. Huang, "Path planning and tracking for vehicle collision avoidance based on model predictive control with multi-constraints," *IEEE Transactions on Vehicular Technology*, vol. PP, no. 99, pp. 1–1, 2016.
[24] G. Galben, "New three-dimensional velocity motion model and composite odometry-inertial motion model for local autonomous navigation," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 3, pp. 771–781, March 2011.
[25] Y. S. Son, W. Kim, S. H. Lee, and C. C. Chung, "Robust multirate control scheme with predictive virtual lanes for lane-keeping system of autonomous highway driving," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 8, pp. 3378–3391, Aug 2015.
[26] U. Muller, J. Ben, E. Cosatto, B. Flepp, and Y. L. Cun, "Off-road obstacle avoidance through end-to-end learning," in *Advances in neural information processing systems*, 2005, pp. 739–746.
[27] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, M. Scoffier, K. Kavukcuoglu, U. Muller, and Y. LeCun, "Learning long-range vision for autonomous off-road driving," *Journal of Field Robotics*, vol. 26, no. 2, pp. 120–144, 2009.
[28] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2722–2730.
[29] X. Xiong, J. Wang, F. Zhang, and K. Li, "Combining deep reinforcement learning and safety based control for autonomous driving," *arXiv preprint arXiv:1612.00147*, 2016.
[30] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
[32] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1–4.
[33] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.
[34] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in neural information processing systems*, 2014, pp. 3320–3328.
[35] S. Song and J. Xiao, "Deep sliding shapes for amodal 3d object detection in rgb-d images," *arXiv preprint arXiv:1511.02300*, 2015.
[36] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

Liangzhi Li received the B.Sc and M.Eng degrees in Computer Science from South China University of Technology (SCUT), China, in 2012 and 2016, respectively. He is currently pursuing the Ph.D. degree in Electrical Engineering at Muroran Institute of Technology, Japan. His main fields of research interest include machine learning, big data, and robotics. He has received the best paper award from FCST 2017.



Kaoru Ota was born in Aizu-Wakamatsu, Japan. She received M.S. degree in Computer Science from Oklahoma State University, USA in 2008, B.S. and Ph.D. degrees in Computer Science and Engineering from The University of Aizu, Japan in 2006, 2012, respectively. She is currently an Assistant Professor with Department of Information and Electronic Engineering, Muroran Institute of Technology, Japan. From March 2010 to March 2011, she was a visiting scholar at University of Waterloo, Canada. Also she was a Japan Society of the Promotion of Science (JSPS) research fellow with Kato-Nishiyama Lab at Graduate School of Information Sciences at Tohoku University, Japan from April 2012 to April 2013. Her research interests include Wireless Networks, Cloud Computing, and Cyber-physical Systems. Dr. Ota has received best paper awards from ICA3PP 2014, GPC 2015, IEEE DASC 2015, IEEE VTC 2016-Fall, FCST 2017 and 2017 IET Communications Premium Award. She is an editor of IEEE Transactions on Vehicular Technology (TVT), IEEE Communications Letters, Peer-to-Peer Networking and Applications (Springer), Ad Hoc & Sensor Wireless Networks, International Journal of Embedded Systems (Inderscience) and Smart Technologies for Emergency Response & Disaster Management (IGI Global), as well as a guest editor of ACM Transactions on Multimedia Computing, Communications and Applications (leading), IEEE Internet of Things Journal, IEEE Access, IEEE Communications Magazine, IEEE Network, IEEE Wireless Communications (2015), IEICE Transactions on Information and Systems (2014), and Ad Hoc & Sensor Wireless Networks (Old City Publishing) (2014). She is the recipient of IEEE TCSC Early Career Award 2017.



Mianxiong Dong received B.S., M.S. and Ph.D. in Computer Science and Engineering from The University of Aizu, Japan. He is currently an Associate Professor in the Department of Information and Electronic Engineering at the Muroran Institute of Technology, Japan. He was a JSPS Research Fellow with School of Computer Science and Engineering, The University of Aizu, Japan and was a visiting scholar with BBCR group at University of Waterloo, Canada supported by JSPS Excellent Young Researcher Overseas Visit Program from April 2010 to August 2011. Dr. Dong was selected as a Foreigner Research Fellow (a total of 3 recipients all over Japan) by NEC C&C Foundation in 2011. His research interests include Wireless Networks, Cloud Computing, and Cyber-physical Systems. He has received best paper awards from IEEE HPCC 2008, IEEE ICESS 2008, ICA3PP 2014, GPC 2015, IEEE DASC 2015, IEEE VTC 2016-Fall, FCST 2017 and 2017 IET Communications Premium Award. Dr. Dong serves as an Editor for IEEE Transactions on Green Communications and Networking (TGCN), IEEE Communications Surveys and Tutorials, IEEE Network, IEEE Wireless Communications Letters, IEEE Cloud Computing, IEEE Access, as well as a leading guest editor for ACM Transactions on Multimedia Computing, Communications and Applications (TOMM), IEEE Transactions on Emerging Topics in Computing (TETC), IEEE Transactions on Computational Social Systems (TCSS). He has been serving as the Vice Chair of IEEE Communications Society Asia/Pacific Region Meetings and Conference Committee, Leading Symposium Chair of IEEE ICC 2019, Student Travel Grants Chair of IEEE GLOBECOM 2019, and Symposium Chair of IEEE GLOBECOM 2016, 2017. He is the recipient of IEEE TCSC Early Career Award 2016, IEEE SCSTC Outstanding Young Researcher Award 2017, The 12th IEEE ComSoc Asia-Pacific Young Researcher Award 2017, and Funai Research Award 2018.