Comfortable Driving by using Deep Inverse Reinforcement Learning

Daiko Kishikawa
Department of Urban Environment Systems
Chiba University
Chiba, Japan
d.kishikawa@chiba-u.jp

Sachiyo Arai
Department of Urban Environment Systems
Chiba University
Chiba, Japan
sachiyo@faculty.chiba-u.jp

***Abstract*—Passenger comfort and their safety are pre-requisites to realizing autonomous driving vehicles. Herein, we define "comfortable driving" by considering "comfortability", with which less physical and mental burden for passengers. Deep reinforcement learning, which has several applications in the autonomous driving domain, is an effective approach to achieve the comfortable driving. Generally, reward function in deep reinforcement learning is expressed quantitatively. However, because obtaining a quantitative expression for comfortable driving is difficult, there is no guarantee that a reward function can satisfy "comfortable driving" conditions. Therefore, we propose an approach to identify reward function that can realize comfortable driving, using LogReg-IRL, a deep inverse reinforcement learning method in linearly solvable Markov decision process. With the constraint that the maximum lateral acceleration does not exceed a certain threshold value, we could experimentally achieve "comfortable driving". Additionally, by calculating the gradient for the state input of the state-dependent reward function, we could analyze important states.**

***Index Terms*—autonomous driving, comfortability, deep inverse reinforcement learning**

I. Introduction

Autonomous driving is a technology used to drive vehicles with little or no human input. An analysis by [1] shows that 90% of traffic accidents are caused by driver error. This suggests that replacing human driver by an autonomous driving agent will be able to reduce traffic-related accidents. In addition to driver safety, it is also important to realize driving that considers the comfort of passenger, such as less physical and mental burden for practical application of autonomous driving technology in manned vehicles. In this study, we define driving that satisfies these requirements as "comfortable driving". At present, commercial autonomous vehicles are controlled by a rule-based program that defines the appropriate controls for all possible situations. From the perspective of achieving robust control, which will enable the vehicle to drive itself in any environment, a rule-based algorithm is inappropriate given that the developers must code all possible scenarios that have to be satisfied.

Recently, reinforcement learning has attracted significant research attention, particularly in the field of autonomous driving. In reinforcement learning, the agent, which is the decision-making entity, learns optimal control through a trial and error process to maximize the cumulative reward. Deep reinforcement learning (DRL) is the combination of reinforcement learning and deep learning. Owing to the exceptional ability of neural networks in feature extraction and function approximation tasks, DRL has achieved promising results in complex tasks including autonomous driving [2]. However, the design of reward function in reinforcement learning is a difficult problem because the reward function must be the most succinct, robust, and transferable definition of the task to be performed [3]. Moreover, there is no guarantee that a comfortable driving experience can be achieved using a reward function based on such a quantitative representation. This may be attributed to the subjective definition of comfortability, which varies across designers and developers.

Thus, to address the above-mentioned problem, inverse reinforcement learning (IRL) [4] has emerged as an effective approach. By observing demonstrations performed by an expert agent that takes intended actions, IRL estimates a reward function that enables the agent in reinforcement learning to imitate the behavior of the expert. Here, "expert agent" means a decision-making entity that takes the desired action to be imitated. Ideally, it is a human driver capable of performing the desired driving. Several IRL algorithms were required to update the reward function in the inner-loop of policy optimization in most of previous studies. Hence, there exist very limited applications of IRL in the domain of autonomous driving because the estimated reward function must be evaluated on a driving simulator using a physical model, or in an actual car in the real world. Sharifzadeh et al. used IRL to learn [5]. They applied apprenticeship learning [3] to deep Q-learning [6]. Our proposed method differs from their methods in the sense that it uses an IRL method without updating the estimation of the reward functions using the inner-loop for achieving comfortable driving.

The remainder of this paper is organized as follows. Section 2 describes preliminaries for the linearly solvable Markov decision process (LMDP) and LogReg-IRL, which is a deep inverse reinforcement learning method in LMDP. Section 3 describes our approach to realize comfortable driving using LogReg-IRL. Section 4 describes the experiments to evaluate the proposed approach and presents the results. Section 5 then analyzes the results obtained from the experiments in the previous section. Finally, section 6 concludes the study and discusses future work.

II. Preliminaries

Let $\mathcal{S}$ and $\mathcal{A}$ represent a state space and an action space, respectively. Then, $P(s'|s,a)$ represents the conditional probability of transitioning to the next state $s' \in \mathcal{S}$ when the action $a \in \mathcal{A}$ is executed in the current state $s \in \mathcal{S}$, and $R(s,a)$ represents the reward when executing the action a in the current state s. In the traditional Markov decision process (MDP) [7], the Bellman equation is expressed as follows:

$$V(s) = \max_a \left\{ R(s,a) + \sum_{s'} p(s'|s,a)\, \gamma V(s') \right\} \tag{1}$$

Here, γ is the discount rate $(0 < \gamma \leq 1)$. In the above, Eq. (1) is non-linear because of the $\max$ operator. Hence, the only approach to find its solution is to use an iterative method such as value iteration.

A. Linearly solvable MDP

Linearly solvable Markov decision process (LMDP) [8] is a subclass of MDP proposed by Todorov. In LMDP, control $\mathbf{u} \in \mathbb{R}^{|\mathcal{S}|}$, which is a real-valued vector with the same dimension as the number of states, is substituted for action a in MDP.

There are two assumptions in LMDP. First, the state transition probability p, i.e., the controlled transition probability, is represented by the product of the uncontrolled transition probability $\bar{p}$ and $\exp(\mathbf{u}_s)$. $\mathbf{u}_s$ is the exponentiated vector of the control about current state s:

$$p(s'|s,\mathbf{u}_s) = \bar{p}(s'|s)\exp(\mathbf{u}_s) \tag{2}$$

The uncontrolled probability $\bar{p}(s'|s)$ is the state transition probability from the current state s to the next state s' when control $\mathbf{u}_s$ is not performed. From Eq. (2), if $\bar{p}(s'|s) = 0$ then $p(s'|s,\mathbf{u}_s) = 0$. Therefore, Eq. (2) can also be written as

$$\begin{aligned} \exp(\mathbf{u}_s) &= \frac{p(s'|s,\mathbf{u}_s)}{\bar{p}(s'|s)} \\ \mathbf{u}_s &= \log\frac{p(s'|s,\mathbf{u}_s)}{\bar{p}(s'|s)} \end{aligned} \tag{3}$$

The second assumption is that the cost $\ell = -R$ is represented by the sum of the state-dependent cost q and the Kullback–Leibler (KL) divergence $D_{\mathrm{KL}}(p|\bar{p})$:

$$\ell(s,\mathbf{u}_s) = q(s) + D_{\mathrm{KL}}\Big(p(s'|s,\mathbf{u}_s)\Big|\Big|\bar{p}(s'|s)\Big) \tag{4}$$

From the definition of KL divergence and Eq. (3), Eq. (4) can be written as

$$\ell(s,\mathbf{u}_s) = q(s) + \sum_{s'} p(s'|s,\mathbf{u}_s)\,\mathbf{u}_s \tag{5}$$

Substituting Eqs. (2) and (5) in Eq. (1), we obtain the following

$$V(s) = q(s) + \min_{\mathbf{u}_s}\left[\sum_{s'} \bar{p}(s'|s)\exp(\mathbf{u}_s)\{\mathbf{u}_s + \gamma V(s')\}\right] \tag{6}$$

Lagrange's undetermined multipliers method is applied to remove the $\min$ operator from the second term in the right-hand side of Eq. (6). Lagrange function $\mathcal{L}$ is defined as

$$\begin{aligned} \mathcal{L}\Big(\mathbf{u}_s, \lambda_s\Big) &= \sum_{s'} \bar{p}(s'|s)\exp(\mathbf{u}_s)\Big(\mathbf{u}_s + \gamma V(s')\Big) \\ &+ \lambda_s\Big(\sum_{s'} \bar{p}(s'|s)\exp(\mathbf{u}_s) - 1\Big) \end{aligned} \tag{7}$$

where the constraint is given by

$$\sum_{s'} \bar{p}(s'|s)\exp(\mathbf{u}_s) = 1 \tag{8}$$

The necessary condition of an extremum with respect to $\mathbf{u}_s$ is as follows

$$\frac{\partial\mathcal{L}}{\partial\mathbf{u}_s} = \bar{p}(s'|s)\exp(\mathbf{u}_s)\Big(\mathbf{u}_s + \gamma V(s')\Big) = 0 \tag{9}$$

$$\frac{\partial\mathcal{L}}{\partial\lambda_s} = \sum_{s'} \bar{p}(s'|s)\exp(\mathbf{u}_s) - 1 = 0 \tag{10}$$

The solution $\mathbf{u}_s^*$ is

$$\mathbf{u}_s^* = -\gamma V(s') - \log\Big(\sum_{s'} \bar{p}(s'|s)\exp\{-\gamma V(s')\}\Big) \tag{11}$$

From Eqs. (6), (8), (11)

$$\begin{aligned} V(s) &= q(s) - \log\Big(\sum_{s'} \bar{p}(s'|s)\exp\{-\gamma V(s')\}\Big) \\ \exp\{-V(s)\} &= \exp\Big\{-q(s)\Big\}\sum_{s'} \bar{p}(s'|s)\exp\{-\gamma V(s')\} \end{aligned} \tag{12}$$

Thus, the $\min$ operator is dropped because it is the minimum. Then, the optimal controlled transition probability p^* is defined as

$$p^*(s'|s) = \frac{\bar{p}(s'|s)\exp\{-\gamma V(s')\}}{\sum_{s'} \bar{p}(s'|s)\exp\{-\gamma V(s')\}} \tag{13}$$

Rearranging terms of Eq. (12), we obtain

$$\begin{aligned} \frac{\exp\{-V(s)\}}{\exp\{-q(s)\}} &= \sum_{s'} \bar{p}(s'|s)\exp\{-\gamma V(s')\} \\ \exp\{q(s) - V(s)\} &= \sum_{s'} \bar{p}(s'|s)\exp\{-\gamma V(s')\} \end{aligned} \tag{14}$$

Substituting Eq. (14) in Eq. (13), the following is obtained

$$\begin{aligned} p^*(s'|s) &= \frac{\bar{p}(s'|s)\exp\{-\gamma V(s')\}}{\exp\{q(s) - V(s)\}} \\ \log\frac{p^*(s'|s)}{\bar{p}(s'|s)} &= -q(s) - \gamma V(s') + V(s) \\ \log\frac{p^*(s,s')}{\bar{p}(s,s')} &= \log\frac{p^*(s)}{\bar{p}(s)} - q(s) - \gamma V(s') + V(s) \end{aligned} \tag{15}$$

From Eq. (15), we can estimate the state-dependent cost $q(s)$ and state value $V(s)$ by using the density ratio estimation method.

B. *Logistic Regression-Based IRL (LogReg-IRL)*

Logistic Regression-Based IRL (LogReg-IRL) [9] is a deep inverse reinforcement learning method without inner-loop, that uses using Eq. (15) and the LogReg [10] criterion, which is the density ratio estimation method using logistic regression. In Eq. (15), there are two density ratios, $p^*(s)/\bar{p}(s)$ and $p^*(s,s')/\bar{p}(s,s')$. When assigning labels $\eta = 1$ and $\eta = -1$ to data that correspond to the optimal controlled transition probability p^* and uncontrolled transition probability $\bar{p}$, respectively, Bayes' theorem can be applied to the first term of the right-hand side in Eq. (15) as follows:

$$\begin{aligned}
\frac{p^*(s)}{\bar{p}(s)} &= \frac{\mathrm{P}(\eta=1|s)\,\mathrm{P}(s)}{\mathrm{P}(\eta=1)}\left\{\frac{\mathrm{P}(\eta=-1|s)\,\mathrm{P}(s)}{\mathrm{P}(\eta=-1)}\right\}^{-1}\\
&= \frac{\mathrm{P}(\eta=1|s)}{\mathrm{P}(\eta=-1|s)}\frac{\mathrm{P}(\eta=-1)}{\mathrm{P}(\eta=1)}\\
\ln\frac{p^*(s)}{\bar{p}(s)} &= \ln\frac{\mathrm{P}(\eta=1|s)}{\mathrm{P}(\eta=-1|s)}\frac{\mathrm{P}(\eta=-1)}{\mathrm{P}(\eta=1)}\\
&= \ln\frac{\mathrm{P}(\eta=1|s)}{\mathrm{P}(\eta=-1|s)} + \ln\frac{\mathrm{P}(\eta=-1)}{\mathrm{P}(\eta=1)} \qquad (16)
\end{aligned}$$

Then, by using a deep neural network $F(s) = f_x(s; w_x)$ with weight w_x, a deep logistic regression classifier can be defined as follows

$$\mathrm{P}(\eta=1|s) = \frac{1}{1+\exp(-F(s))} \qquad (17)$$

$$\mathrm{P}(\eta=-1|s) = \frac{1}{1+\exp(F(s))} \qquad (18)$$

Substituting Eqs. (17), (18) in the first term of the right-hand side in Eq. (16), we obtain

$$\begin{aligned}
\ln\frac{\mathrm{P}(\eta=1|s)}{\mathrm{P}(\eta=-1|s)} &= \ln\frac{1+\exp(F(s))}{1+\exp(-F(s))}\\
&= \ln\frac{\exp(F(s))(1+\exp\{-F(s)\})}{1+\exp\{-F(s)\}}\\
&= \ln\exp(F(s))\\
&= F(s) \qquad (19)
\end{aligned}$$

By defining "data D^* that corresponds to the optimal controlled transition probability p^*" as the **expert** data and "data $\bar{D}$ that corresponds to the uncontrolled transition probability $\bar{p}$" as the **baseline** data, the second term of the right-hand side in Eq. (16) can be estimated by $N_{\bar{D}}/N_{D^*}$, which is the ratio of number of baseline data $N_{\bar{D}}$ and number of expert data N_{D^*}. Thus, the ratio $p^*(s)/\bar{p}(s)$ can be estimated as follows:

$$\ln\frac{p^*(s)}{\bar{p}(s)} = F(s) + \ln\frac{N_{\bar{D}}}{N_{D^*}} \qquad (20)$$

The weight w_x of deep logistic regression classifier $C(\eta|s) = (1+\exp\{-\eta F(s)\})^{-1}$ can be estimated by the regularized negative log-likelihood loss function, as shown below:

$$\begin{aligned}
J(w_x) = &- \frac{1}{N_{\bar{D}}}\sum_{j=1}^{N_{\bar{D}}}\ln\{C(\eta=-1|\bar{s}_j)\}\\
&- \frac{1}{N_{D^*}}\sum_{i=1}^{N_{D^*}}\ln\{C(\eta=1|s_i^*)\}\\
&+ \frac{\lambda_x}{2}||w_x||_2 \qquad (21)
\end{aligned}$$

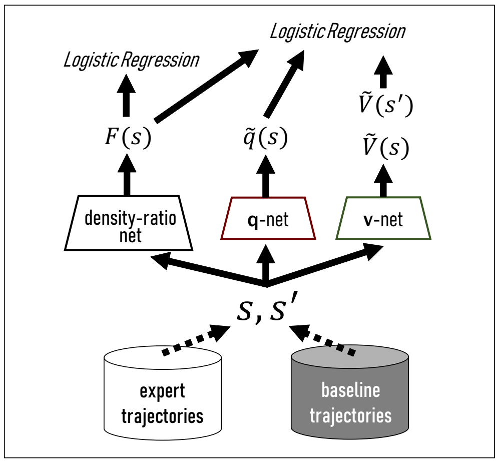

Fig. 1. Training deep neural networks in LogReg-IRL.

The last term of the right-hand side in Eq. (21) is the L2-regularization term with the constant λ_x.

Next, the second ratio $\frac{p^*(s,s')}{\bar{p}(s,s')}$ can be defined from Eqs. (15), (20) as

$$\log\frac{p^*(s,s')}{\bar{p}(s,s')} = F(s) - \tilde{q}(s) - \gamma\tilde{V}(s') + \tilde{V}(s) + \ln\frac{N_{\bar{D}}}{N_{D^*}} \qquad (22)$$

where $\tilde{q}(s) = f_q(s; w_q)$ and $\tilde{V}(s) = f_V(s; w_V)$ are two deep neural networks that estimate the state-dependent costs $q(s)$ and state value $V(s)$, respectively. We define the deep logistic regression classifier $C(\eta|s,s')$ as

$$C(\eta|s,s') = \frac{1}{1+\exp(-\eta\{F(s)-\tilde{q}(s)-\gamma\tilde{V}(s')+\tilde{V}(s)\})} \qquad (23)$$

such that weights w_q and w_V can be estimated using Eqs. (21), (23).

III. Approach

In this section, we describe an approach to achieve comfortable driving using LogReg-IRL. Our approach consists of the following two steps.

A. *Estimating state-dependent cost and state value for comfortable driving by LogReg-IRL*

First, we train three networks in LogReg-IRL; the density ratio network $F(s)$, state-dependent cost network $\tilde{q}(s)$, and state value network $\tilde{V}(s)$. As shown in Fig. 1, training uses current state s and next state s', which are sampled randomly from expert trajectories that satisfy conditions for comfortable driving and baseline trajectories that do not satisfy these conditions.

B. *Achieving comfortable driving using shaped reward*

Then, we calculate shaped reward $r(s,s')$, which is based on the theory of reward shaping [11], by using the estimated $\tilde{q}(s)$ and $\tilde{V}(s)$ as follows

$$r(s,s') = -\left\{\tilde{q}(s) + \gamma\tilde{V}(s') - \tilde{V}(s)\right\} \qquad (24)$$

Here, r is used for training the agent in reinforcement learning as shown in Fig. 2.

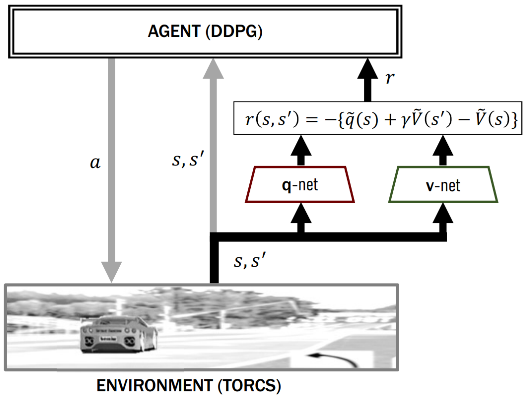
AGENT (DDPG)
a
s, s'
r
$r(s,s') = -\{\tilde{q}(s) + \gamma\tilde{V}(s') - \tilde{V}(s)\}$
q-net
v-net
s, s'
ENVIRONMENT (TORCS)

Fig. 2. Training of autonomous driving using shaped reward.

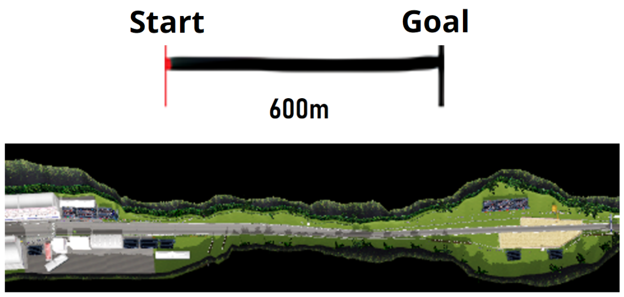

Fig. 3. Course of TORCS used in this experiment.

IV. Experiments

A. Experiment settings

The proposed method was validated using a straight driving experiment. We used Deep Deterministic Policy Gradient (DDPG) [12] in TORCS (The Open Racing Car Simulator) [13] , which is an open source driving simulator. In this validation, we utilize the 600-m straight driving task that uses the course shown in Fig. 3.

A total of 31 sensors were used as state inputs (shown in Table 1) from among 79 dimensions prepared in TORCS. To handle straight driving in the experiment, one-dimensional steering operation was used as an action output.

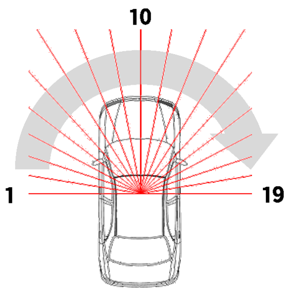

Fig. 4. track(19 dim.); red lines represent sensors.

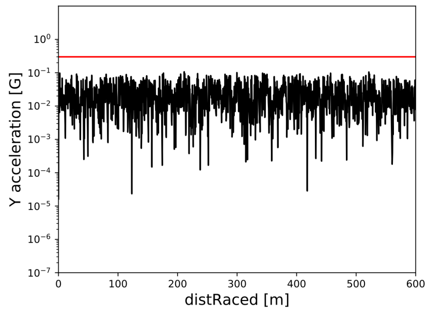

Fig. 5. Lateral acceleration in expert trajectory.

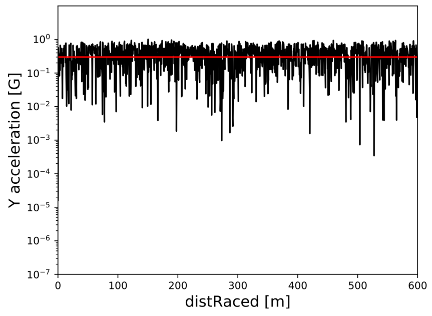

Fig. 6. Lateral acceleration in baseline trajectory.

From an experimental perspective, for the condition of comfortable driving, the maximum allowed lateral acceleration during driving was less than 0.3 g. This was based on a study [14] that suggested that passengers experience discomfort or fear when the lateral acceleration exceeds 0.3 g.

B. Generating trajectories

First, we trained the DDPG agent using the reward function developed in a previous study [15]. By learning 7000 episodes, ten models were obtained that satisfied the condition of comfortable driving. For each model obtained, two trajectories were generated by adding noise of different ranges of uniform distribution to the action output; ± 0.01 for expert trajectories and ± 0.2 for baseline trajectories. Figs. 5 and 6 illustrate an example lateral acceleration when traveling

TABLE I
THE 31 SENSORS USED IN THIS STUDY.

sensor	range	unit	description
angle	$[-\pi, +\pi]$	[rad]	angle of car
gear	$-1, 0, 1, 2, 3, 4, 5, 6$	-	current gear
rpm	$[0, +\infty)$	[rpm]	number of rotations of engine
speedX speedY speedZ	$[-\infty, +\infty]$ $[-\infty, +\infty]$ $[-\infty, +\infty]$	[km/h] [km/h] [km/h]	X-axis speed Y-axis speed Z-axis speed
track1-19 (19 dim.)	$[0, 200]$	[m]	distance to wall (Fig. 4)
trackPos	$[-1+1]$	-	displacement from center of road
wheelSpinVel (4 dim.)	$[0, +\infty]$	[rad/s]	speed of four wheels
z	$[-\infty, +\infty]$	[m]	displacement from center of gravity

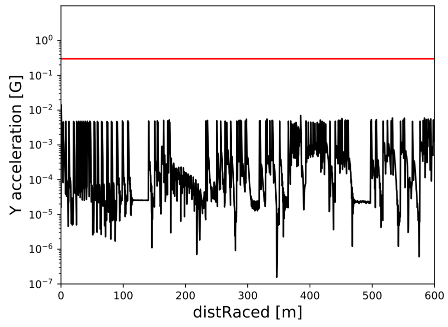

Fig. 7. Lateral acceleration achieved during driving.

in expert trajectory and baseline trajectory. In Figs. 5 and 6, the horizontal axis ("distRaced") indicates the distance travelled, the vertical axis ("Y acceleration") indicates the lateral acceleration, and the red line indicates the condition of 0.3 g.

C. Training networks of LogReg-IRL

Next, the networks of LogReg-IRL were trained using the generated trajectories. Multilayer perceptron, which comprises one input layer, one output layer, and two hidden layers, was used as the network. The number of inputs and outputs of each layer was (31, 24), (24, 12), (12, 6), and (6, 1). We used a scaled exponential linear unit [16] as the activation function. To prevent over-fitting, we applied dropout with probabilities of 20% and 50% to the input layer and the hidden layer, respectively.

D. Learning automatic driving using shaped reward

Finally, we trained the DDPG agent in the TORCS environment by using the shaped reward function $r(s, s')$, which is obtained by using estimated $\tilde{q}(s)$ and $\tilde{V}(s)$ from Eq. (24). The lateral acceleration of the model obtained by r is shown in Fig. 7. In Fig. 7, the horizontal axis indicates the travel distance, the vertical axis indicates the lateral acceleration, and the red line indicates the condition of 0.3 g. The maximum lateral acceleration of 0.01 g during satisfied the set conditions for comfortable driving. Therefore, LogReg-IRL was able to estimate the reward function that can achieve comfortable driving.

V. Analyses

A. Analysis of the density ratio network in LogReg-IRL

We examined how the density ratio network $F(s) = f_x(s; w_x)$, which plays an important role in learning of LogReg-IRL, was trained. Using $F(s)$ as a deep logistic regression classifier $C(\eta|s) = (1 + \exp\{-\eta F(s)\})^{-1}$, we tested whether the expert and baseline trajectories generated using the approach described in in Section 4.2 were properly classified. The results are summarized in Table II. We evaluated the performance of the classifier based on the following seven indices.

TABLE II
Confusion matrix.

		True		sum
		Expert	Baseline	
Predicted	Expert	80131	444	80575
	Baseline	52474	141963	194437
sum		132605	142407	275012

TABLE III
Calculated values of the seven indices.

Index	Value
Accuracy	80.76%
Recall	60.43%
Precision	99.45%
Specificity	73.01%
F-score	75.18%
False baseline rate	27.00%
False expert rate	0.55%

- **Accuracy** $\mathrm{P}_{ac} \cdots$ Probability of classifying an expert as an expert and a baseline as a baseline.
- **Recall** $\mathrm{P}_{re} \cdots$ Probability of classifying an actual expert as an expert.
- **Precision** $\mathrm{P}_{pr} \cdots$ Probability of being actually an expert among data classified as an expert.
- **Specificity** $\mathrm{P}_{sp} \cdots$ Probability of being actually baseline among data classified as baseline.
- **F-score** $f \cdots$ Harmonic mean of recall and precision.
- **False baseline rate** $\mathrm{P}_{\tilde{b}} \cdots$ Probability of classifying an expert as a baseline.
- **False expert rate** $\mathrm{P}_{\tilde{\pi}} \cdots$ Probability of classifying a baseline as an expert.

Let $N_{\tilde{\pi}=\pi}$ be the number of experts classified as experts, $N_{\tilde{\pi}=b}$ be the number of baselines classified as experts, $N_{\tilde{b}=\pi}$ be the number of experts classified as baselines, and $N_{\tilde{b}=b}$ be the number of baselines classified as baselines. Then, each index can be estimated using the following :

$$\begin{aligned}
\mathrm{P}_{ac} &= \frac{N_{\tilde{\pi}=\pi} + N_{\tilde{b}=b}}{N_{\tilde{\pi}=\pi} + N_{\tilde{\pi}=b} + N_{\tilde{b}=\pi} + N_{\tilde{b}=b}} \\
\mathrm{P}_{re} &= \frac{N_{\tilde{\pi}=\pi}}{N_{\tilde{\pi}=\pi} + N_{\tilde{b}=\pi}}, \mathrm{P}_{pr} = \frac{N_{\tilde{\pi}=\pi}}{N_{\tilde{\pi}=\pi} + N_{\tilde{\pi}=b}} \\
\mathrm{P}_{sp} &= \frac{N_{\tilde{b}=b}}{N_{\tilde{\pi}=b} + N_{\tilde{b}=b}} \\
f &= \frac{2}{\frac{1}{\mathrm{P}_{re}} + \frac{1}{\mathrm{P}_{pr}}} \\
\mathrm{P}_{\tilde{b}} &= \frac{N_{\tilde{b}=\pi}}{N_{\tilde{\pi}=\pi} + N_{\tilde{b}=\pi}}, \mathrm{P}_{\tilde{\pi}} = \frac{N_{\tilde{\pi}=b}}{N_{\tilde{\pi}=b} + N_{\tilde{b}=b}}
\end{aligned}$$

Table III lists the values of the seven indices calculated from the confusion matrix. In general, there exists a trade-off relationship between the precision rate and the recall rate. From the results in Table III, the accuracy rate is extremely high compared with the recall rate, and the false expert rate is extremely low. Therefore, learning can be said to have been performed such that the baseline was not erroneously classified as an expert.

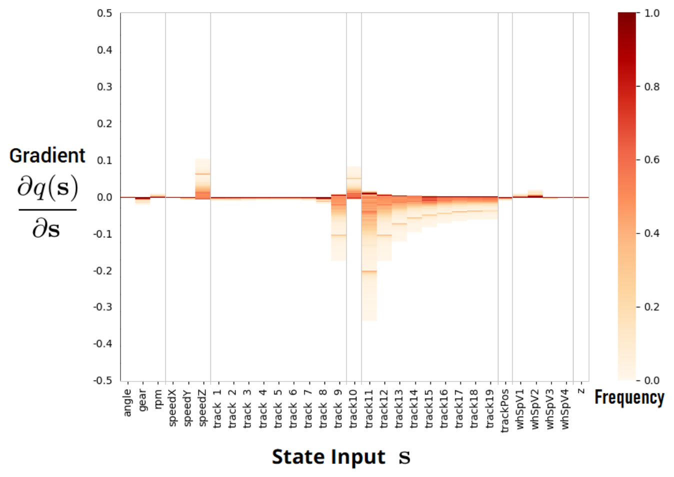

Fig. 8. Histogram of the gradient of $\tilde{q}(s)$ with respect to the state input s.

B. Analysis of the gradient of $\tilde{q}(s)$

Next, for the state-dependent cost network $\tilde{q}(s)$, we calculated the gradient $\partial\tilde{q}(s)/\partial s$, which is defined as the amount of change in the output value of $\tilde{q}(s)$ with respect to slight changes in the state input s, and investigated its distribution. It was found that the larger the value of the gradient, the greater the change in the reward for the change in the state input; in other words, for larger values of gradient, a stronger effect on the reward can be expected. The results are shown in Fig. 8.

We observed a tendency, wherein a positive gradient is distributed along "speedZ", which is the speed in the Z direction, and "track10", which is the distance between the vehicle and the end of the course with respect to the front direction. This tendency will increase the distance between the front end and the end of the course, thereby reducing the possibility of course-out. Moreover, "speedZ" can also reduce the possibility of the course-out as it avoids deceleration along the Z-direction, which occurs when going out of course. We also observed a tendency for negative gradients to be distributed in "track9", which is the distance to the course end 10 degrees to the left, with respect to the front direction. A similar tendency was also observed between "track11" to "track19", which is the distance to the right course end with respect to the front direction. This distribution may be attributed to the fact that the reward corresponding to the position of the vehicle was estimated from the distance to the course end on the right side. The course used in the experiment has a gentle Z-shaped curve. Therefore, if the agent does not steer the vehicle, the vehicle will collide with the left wall of the course in the first half of the 600 m section. Therefore, the reward to steer to the right of the course was estimated accordingly.

VI. Conclusion

In this study, we focused on defining the reward function that can realize comfortable driving in reinforcement learning. Therefore, we proposed a novel method to realize safe driving by using the reward estimated by LogReg-IRL. The results of the experiments suggest that we were able to achieve comfortable driving using LogReg-IRL. Further, by analyzing the gradient of the estimated state-dependent cost $\tilde{q}(s)$, we analyzed the state inputs that are affected when calculating the reward. Although the condition of comfortable driving (maximum lateral acceleration less than 0.3 g) was experimentally set, this condition can be arbitrarily set as long as a trajectory satisfying the condition can be generated.

Unlike several previous inverse reinforcement learning methods, in addition to expert trajectories, LogReg-IRL also requires baseline trajectories that do not satisfy the condition. Moreover, the learning performance of LogReg-IRL is greatly dependent on baseline trajectories, which is a major concern. As future work, we intend to develop a method that is less affected by the property of the baseline trajectory.

References

[1] National Highway Traffic Safety Administration, "Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey," 2015.
[2] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, and A. Shah, "Learning to Drive in a Day," arXiv preprint arXiv:1807.00412, 2018.
[3] P. Abbeel, A. Y. Ng, "Apprenticeship Learning via Inverse Reinforcement Learning," Proceedings of the 21st International Conference on Machine Learning, pp. 1-8, 2004.
[4] A. Y. Ng, S. Russell, "Algorithms for Inverse Reinforcement Learning," Proceedings of the 17th International Conference on Machine Learning, pp. 663-670, 2000.
[5] S. Sharifzadeh, I. Chiotellis, R. Triebel, and D. Cremers, "Learning to Drive using Inverse Reinforcement Learning and Deep Q-Networks," arXiv preprint arXiv:1612.03653, 2016.
[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg and D. Hassabis, "Human-level control through deep reinforcement learning," Nature 518 (7540), pp. 529-533, 2015.
[7] R. Bellman, "Dynamic Programming," Princeton University Press, 1957.
[8] E. Todorov, "Linearly-solvable Markov decision problems," Advances in Neural Information Processing Systems, pp. 1369-1376, 2006.
[9] E. Uchibe, "Model-Free Deep Inverse Reinforcement Learning by Logistic Regression," Neural Processing Letters, Vol. 47, Issue 3, pp. 891-905, 2017.
[10] S. Bickel, M. Brückner, and T. Scheffer, "Discriminative Learning for Differing Training and Test Distributions," Proceedings of the 24th International Conference on Machine Learning, pp. 81-88, 2007.
[11] A. Y. Ng, D. Harada, and S. Russell, "Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping," Proceedings of the 16th International Conference on Machine Learning, Vol. 99, pp. 278-287, 1999.
[12] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," arXiv preprint arXiv:1509.02971, 2015.
[13] B. Wymann, et al. , "TORCS, The Open Racing Car Simulator," http://torcs.sourceforge.net.
[14] K. Nasukawa, Y. Miyashita, and M. Shiokawa, "Efficiency Tests for Running: Third Revised Edition," Sankaido Publishing, p. 54, 1999.
[15] S. Kitamura, S. Ishikawa, and S. Arai, "Filtering Environmental Information for Automatic Driving in Urban Area," Proceedings of the 32nd Annual Conference of the Japanese Society for Artificial Intelligence, 2018.
[16] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-Normalizing Neural Networks," In Advances in Neural Information Processing Systems, pp. 971-980, 2017.