Extracting Human-Like Driving Behaviors From Expert Driver Data Using Deep Learning

Kyle Sama, Yoichi Morales, *Member, IEEE*, Hailong Liu, *Member, IEEE*, Naoki Akai, *Member, IEEE*, Alexander Carballo, *Member, IEEE*, Eijiro Takeuchi, *Member, IEEE*, and Kazuya Takeda, *Senior Member, IEEE*

***Abstract*—This paper introduces a method to extract driving behaviors from a human expert driver which are applied to an autonomous agent to reproduce proactive driving behaviors. Deep learning techniques were used to extract latent features from the collected data. Extracted features were clustered into behaviors and used to create velocity profiles allowing an autonomous driving agent could drive in a human-like manner. By using proactive driving behaviors, the agent could limit potential sources of discomfort such as jerk and uncomfortable velocities. Additionally, we proposed a method to compare trajectories where not only the geometric similarity is considered, but also velocity, acceleration and jerk. Experimental results in a simulator implemented in ROS show that the autonomous agent built with the driving behaviors was capable of driving similarly to expert human drivers.**

***Index Terms*—Autonomous driving, autoencoder, driving behavior, deep learning.**

I. Introduction

In RECENT years, there has been increased interest in Autonomous Driving. Large corporations, such as Waymo, Tesla, Cruise, and Uber, are investing heavily into it [2]. Traditional model-based techniques for autonomous driving vehicles use maps to localize themselves, plan and follow paths to their destination while respecting traffic rules [3], [4]. Current high definition environmental maps contain the road lane network, traffic sign locations and semantic information, such as traffic rules. These maps make it possible to compute paths for autonomous vehicles, however, in many cases they do not convey all the information required for a comfortable ride.

Passenger comfort, which has been a focus of recent research [5], is not necessarily attained by strictly following traffic rules. One factor involved in comfort is risk perception [6], which is a passenger's ability to identify potential dangers and risky behavior. For example, in certain situations, driving with only regards to traffic rules can in fact be dangerous. Driving at the speed limit through an unsignaled intersection, even with right of way, could lead to a situation where the driver cannot stop in time to avoid a suddenly appearing obstacle. Drivers tend to make a trade-off between safety, speed and ride comfort [7], which is difficult to model without using human data. One way to model this trade-off is to extract proactive behavior from expert drivers; people who drive safely, defensively anticipating potential dangers and proactively acting in a way to mitigate them [8]. Proactive meaning that these drivers take measures to protect themselves, suspecting potential oncoming hazards that are not immediately visible.

Safe driving behavior at intersections is especially important since much of the difference in driving behavior is related to handling various types of intersections [9]. Over 50% of all traffic accidents in Japan occur at or near intersections [10]. In the US, 22% of fatalities and 45% of injuries in crashes are intersection-related [11]. Pedestrian collisions in the US resulted, 98.6% of the time, in pedestrian injury or fatality [12]. These accidents are due to a driver's inability to both assess and predict potential dangers in intersections [13]. The significant number of accidents and subsequent risk of death and injury illustrate the importance of safe or proactive driving at intersections.

In this work, the target environments are typical residential areas in Japan; characterized by narrow roads, no traffic lights, and few road markings. In these areas, it is common to find unmarked unsignaled intersections with blind corners, which are challenging for self-driving cars, since on-board sensors cannot detect obstacles in occluded areas [14].

In this work we propose to create an autonomous agent from driving behaviors extracted from expert driver data. This agent will then be compared using our trajectory comparison method.

Manuscript received September 2, 2019; revised January 10, 2020; accepted February 18, 2020. Date of publication March 16, 2020; date of current version October 13, 2020. This work was supported by the Center of Innovation Program (Nagoya-COI) from Japan Science and Technology Agency. This article is an extension of a paper presented at the 21st International Conference on Intelligent Transportation Systems, Maui, HI, USA, Nov. 2018. The review of this article was coordinated by the Guest Editors of the Special Section on Connected and Autonomous Vehicles. *(Corresponding author: Kyle Sama.)*

Kyle Sama, Hailong Liu, and Naoki Akai are with the Graduate School of Information Science, Nagoya University, Nagoya 463-8603, Japan (e-mail: sama.kyle@g.sp.m.is.nagoya-u.ac.jp; lhl881210@gmail.com; akai@coi.nagoya-u.ac.jp).

Yoichi Morales is with the Institute of Innovation for Future Society, Nagoya University, Nagoya 464-8603, Japan (e-mail: yoichims@ieee.org).

Alexander Carballo is with the Institute of Innovation for Future Society, Nagoya University, Nagoya 464-8603, Japan, and also with the TierIV Inc., Nagoya University Open Innovation Center, Nagoya 450-6610, Japan (e-mail: alexander@g.sp.m.is.nagoya-u.ac.jp).

Eijiro Takeuchi is with the Graduate School of Information Science, Nagoya University, Nagoya 463-8603, Japan, and also with the TierIV, Inc., Nagoya University Open Innovation Center, Nagoya 450-6610, Japan (e-mail: takeuchi@coi.nagoya-u.ac.jp).

Kazuya Takeda is with the Takeda Laboratory, Graduate School of Information Science, Nagoya University, Nagoya 463-8603, Japan, and also with the TierIV, Inc., Nagoya University Open Innovation Center, Nagoya 450-6610, Japan (e-mail: kazuya.takeda@nagoya-u.jp).

Digital Object Identifier 10.1109/TVT.2020.2980197

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

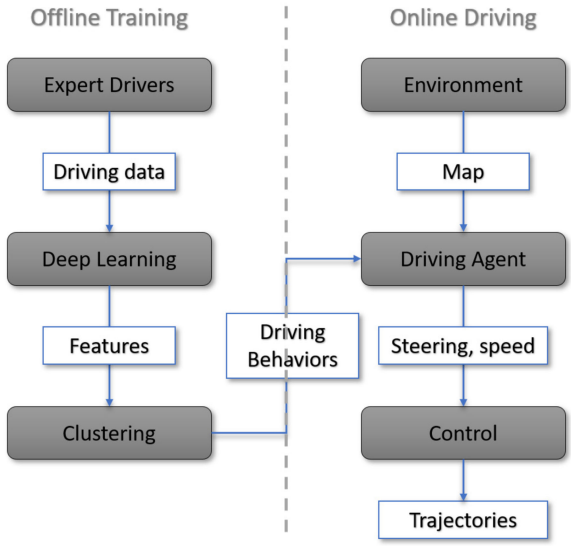


Fig. 1. Driving data is collected from expert drivers and features were extracted using an autoencoder. Similar features were clustered together to create driving behaviors. The behaviors were passed, along with an environmental vector map to a driving agent. The agent then computed steering and velocity commands resulting in safe and comfortable trajectories similar to human driven ones.

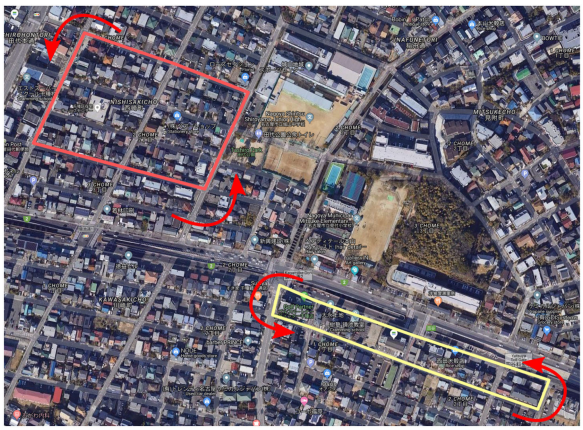


Fig. 2. Top view of a typical residential area featuring different types of intersections in Nagoya, Japan. The paths in red show the areas used to train the autoencoder (Section III-C). The yellow paths in the bottom part show the areas used to test the encoders.[1] Both paths move in a counterclockwise direction.

The flow of the work is shown in Fig. 1. First we took driving data from experienced proactive drivers, as well as elderly drivers. They drove continuously in suburban residential roads involving turns, braking and acceleration at various types of intersections (4-way with and without stops, T-intersections, etc.) and moving through various straight segments of road[1] (Fig. 2). We used autoencoders to perform driving data dimension reduction, extracting representative feature vectors from the input data set. We clustered these driving features, resulting in distinct driving behaviors. The driving behaviors, together with a map, were fed to a driving agent which was able to produce steering and velocity commands which resulted in trajectories similar to the human expert driven ones. Details are given in the following sections.

The contributions of this work are:

- a data-driven approach with deep autoencoders to extract driving behaviors which are used in a driving agent which can compute trajectories similar to human expert drivers.
- showing it is possible to leverage the collected data to learn how to drive in a human-like fashion, safely and comfortably, in the absence of obstacles.
- a new method to score trajectory comparisons.

The structure of the work is as follows: In Section II we discuss relevant works related to our research and how we address some of the problems remaining unsolved within the realm of proactive driving and feature extraction. In Section III we explain our methodology of using feature extraction to extract driving behaviors. Next, in Section IV we introduce our experimental setup, detailing how we collected and used data to illustrate our methods. With Section V we present the results of those experiments. In Section VI and Section VII we provide final discussion and our conclusions, respectively.

II. Related Works

This section details related works concerning proactive driving, feature extraction, driving policies and works based on simulators. Additionally, it also provides a discussion on the differences between other works and the approach presented in this paper.

A. Proactive Driving

Proactive or defensive driving in order to avoid potential hazards is an important topic for traffic and autonomous driving research. Yoshihara *et al.* illustrates the importance of modeling driving behavior for dangerous situations, such as in blind intersections where vision is partially or completely obscured [8]. There are works which describe computational solutions for autonomous vehicles when navigating areas with high a likelihood of collisions with other vehicles, pedestrians and cyclists [15]. Morales *et al.* computed visibility through an index [16] and created a trajectory planner which considers occluded and non-occluded features of the environment [17].

Another way to tackle safety issues could be to leverage big data or user connectivity in co-operative networks, such as in [18] where they illustrate how densely connected users on a network can be clustered in order to deliver efficient and secure data transmission. In a similar vein, nearby drivers, potentially using vehicular ad hoc networks (VANETs), could share their location; alerting those at intersections of potential oncoming hazards [19].

Examining the driving performance of elderly drivers indicate that they make more errors than average drivers. Infrequent drivers, especially amongst the elderly, would indicate similar results as well as show the need for further driving education [20]. Elderly drivers are also over-represented in crashes occurring at intersections [21], and are more at risk of fatal

[1]The experimental environment can be seen in detail in the following URL: [Online]. Available: https://drive.google.com/open?id=1f6xpYsBtWAy3VHIfocn3WwOlX8VB-ECO&usp=sharing

crash involvement in intersection and other related crashes [22]. A potential solution for this increase in crash involvement due to age, or required training for infrequent or inexperienced drivers, could come from the study and development of autonomous agents based on expert driving data.

Differently from previous works involving proactive driving [23], our work focuses on using deep learning to extract proactive behaviors exhibited by expert drivers.

B. Feature Extraction

In previous works we have proven that we can extract these proactive behaviors from expert driver data and create driving models from them [23], [24]. Using machine learning and handpicked features, we created an agent that could navigate through intersections pro-actively. The problem is determining the validity of our handpicked features and whether they are accurate or even relevant. Defining features in this way is impractical due to the need for specifying specific parameters in each situation and requiring robust domain knowledge to make informed choices. Each additional feature rapidly increases complexity of the system, limiting the total number of features that can be used.

Using feature extraction has been used by Li *et al.* [25] to study and improve driving behavior; teaching low-skilled drivers by learning from high-skilled ones. Different types of data can be used for feature extraction, such as GPS data in Gao *et al.* [26] and, like in Al-Sahaf *et al.* [27], they used genetic programming to extract features. Liu *et al.* proposed methods in [28] and [29] for feature extraction involving dimension reduction using a deep sparse autoencoder. They introduce the color map concept for visualization of the extracted features. Yang *et al.* [30] show that both deep learning techniques and camera data can be used for feature extraction. The work by Yurtsever *et al.* [31] uses multiple levels of autoencoders to extract driving signals from different drivers and then uses them in a feed-forward network to predict driving behavior for traffic simulation. Different from other works, our work on unsupervised feature extraction goes further by using the extracted features to generate behavioral models for creating autonomous agents.

C. Driving Policies and Simulators

There are many ways to model driving behavior, such as the "intelligent driver model" (IDM) [32]. Other methods attempt to learn how to drive from human drivers themselves [33]. Some works use handpicked features and use an inverse reinforcement learning framework to learn driving policies from human data, to be used in autonomous vehicles [34], [35]. Bansal *et al.*, with their work titled "ChauffeurNet" [36], shows a novel way to model behavior by representing the ego vehicle, traffic and the environment, with a 2D top down perspective. A series of flat images represent the various input (ego and other vehicle poses, traffic light color and positions, speed limits, routes and road-maps) and produces output images with a new pose and a future trajectory. Even with state of the art learning-based driving agents, such as with the ChauffeurNet project, the results are still not as effective as motion planning based approaches. However, these learning-based methods are essentially trying to implicitly model human experience. The kind of experience that only comes from living and driving for many years. Differently, in this work, we use unsupervised feature extraction techniques on data collected from skilled drivers to create accurate driving models from the extracted behaviors. An agent is then created that can navigate through a suburban environment imitating an expert driver's behavior.

III. Proposed Approach

This section presents the approach used to extract driving behaviors. The definition and composition of driving behavior is presented. A discussion and the concepts of safe and comfortable navigation are provided. An explanation of the use of a deep autoencoder to extract driving behaviors is included, and a trajectory comparison method is presented.

A. Driving Behavior

Driving behavior is the conduct of a driving agent as it navigates roads in order to reach its destination; it is composed of driving actions. Driving actions are the smallest definable elements of our driving infrastructure. They are discrete and performed by a driving agent depending on internal factors (such as preferred velocity and acceleration), external environmental factors, and the agent's destination. Examples of driving actions are: stop, slow down, and accelerate. Driving actions can be parameterized by defining their attributes, such as magnitude of speed and acceleration. Therefore, driving behaviors are a set of one or more parameterized actions describing a particular maneuver that a driving agent performs. "Accelerating slowly from a stopped position," "a sharp left turn" or "maintaining a high speed" are examples of behaviors. This paper aims to extract these behaviors and how they are performed in order to incorporate them into a driving agent.

To extract driving behaviors that result in a safe and comfortable ride for the passengers, we used the data from human expert drivers who can drive smoothly while proactively avoiding potential hazards [24]. These experts have 10+ years of driving experience as well as professional training. Our approach modeled these proactive driving behaviors which are difficult to model solely with topographical maps, vector maps and traffic regulation information.

The data we used was selected to cover a wide variety of scenarios encountered in a typical urban environment (e.g., different configurations of intersections), in order to observe the driver reactions in different situations. As opposed to using a motion planning model, using driving feature extraction can impart desirable qualities from experts, into an autonomous agent.

B. Safe and Comfortable Driving

Safe driving implies that a vehicle arrives at its destination collision-free and out of danger. To do this means that the vehicle should follow basic safety and traffic laws such as obeying speed limits, staying within lane markers and following traffic signs. Comfort, unlike safety, does not have strict laws and regulations

defining what it should be. We define a comfortable drive as providing an experience that puts passengers at ease and relaxed while in the vehicle, while avoiding situations causing unease or discomfort.

While it is difficult to measure an abstract concept such as comfort, it can be described as the lack of discomfort; where the main method to obtain comfort is to avoid sources of discomfort [37]. There are several ways we can attempt to quantify sources of discomfort, or at least explain what might cause them. An obvious first would be the agent not following the aforementioned traffic laws and regulations, since breaking them could cause obvious discomfort by subverting rules aimed at maintaining safety. But within the bounds of respecting those laws and regulations, human drivers can still have different behaviors. These varying practices by individual drivers, such as preferred speed and acceleration styles, will have an effect on passengers feeling safe and comfortable.

There are several things which affect comfort negatively: jerk, lateral-motion and risk perception. Jerk is a measure of the change in acceleration. Even a small amount of it is uncomfortable for most humans and can be a source of motion sickness [38]. Lateral-motion or "zig-zagging" causes unnecessary horizontal acceleration that can also cause discomfort. Risk perception reflects a passenger's experience with potential traffic hazards. This perception is affected by an individual's experiences, societal and personal factors. Low risk perception has a significant relationship with road accidents [6]. Risky situations will cause discomfort in passengers with a high perception of risk. All these factors can lead to an uncomfortable ride for passengers.

Potentially reckless drivers may accelerate up to the speed limit quickly or brake hard from high speeds while overly cautious drivers may advance into a roadway slowly or frequently come to a stop while trying to merge. Both behaviors could be seen as uncomfortable, or even dangerous, to passengers within the vehicle. In addition, while a driver can technically move at the allowed maximum speed through a low visibility intersection, passengers may not feel comfortable in this situation. It may be difficult for the vehicle to fully stop in time should the need arise. For example, when a pedestrian, cyclist or another vehicle becomes suddenly visible and is approaching the intersection from behind a blind corner. Instead, by driving proactively and thus avoiding these situations, an agent will not cause these feelings of discomfort.

In particular, this work aims at limiting the amount of discomfort due to velocity, lateral acceleration and jerk by having our agent imitate expert drivers. The proactive driving behaviors performed by these experts mitigate these causes.

C. Deep Autoencoder

Section III-A mentioned that certain expert drivers may exhibit ideal driving behaviors which incorporate these ideas of safety and comfort by default. In this work we use deep learning to extract these behaviors.

An autoencoder is built from two symmetric deep-belief feed forward networks, each consisting of fully connected neural layers (Fig. 3). These deep networks typically use around five layers in the encoder and decoder each [39]. This network is trained to reconstruct the input as the output. Deep networks offer the advantage that a network with hidden layers can approximate any mapping arbitrarily well [40]. An ideal autoencoder encodes input vector $\vec{i}$ and decodes as output $\vec{i}$.

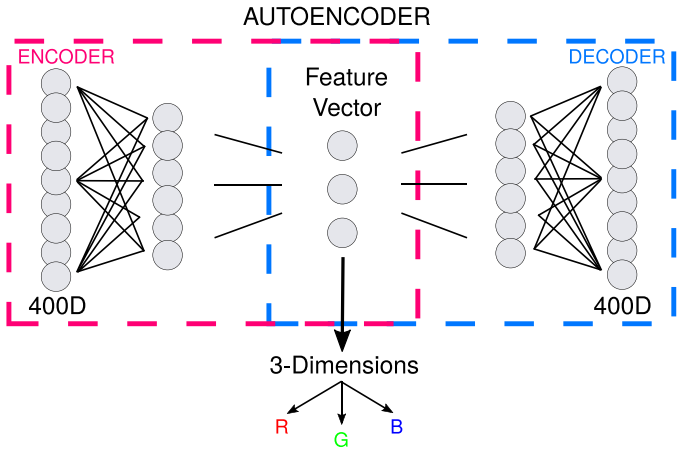


Fig. 3. An architectural diagram of the autoencoder network trained on the driving data. The encoder section receives input and reduces the dimensionality of the data until it reaches the desired dimension. The decoder section reconstructs the input from the feature vector. A 3-dimension feature vector allows it to be easily converted to an RGB value, which aids in visualization. The input data is a 400-dimension vector (as described later in Section IV-B), and the feature vector ends up being in 3 dimensions.

$$g(f(\vec{i})) = \vec{i} \tag{1}$$

f and g are the encoding and decoding functions respectively. The encoding function output is a feature vector $\vec{h}$ representing the latent features extracted from the input data.

$$f(\vec{i}) = \vec{h} \tag{2}$$

These latent features of vector $\vec{h}$ are used to express our extracted driving behaviors, which are used in the next steps of the data processing pipeline.

The input data consists of many dimensions, and they can be shown by a vector:

$$\vec{y} \in \mathbb{R}^d$$

To add a sense of history to our input vector, or a "memory" of the past few seconds, we performed a rolling window. The operation takes N_y consecutive time steps from vector $\vec{y}$ at intervals of t seconds for a total window of $N_y t$ seconds. In this way, each vector has past state information about the vehicle's trajectory. Multiplying this window by the number of chosen features, we end up with an input vector $\vec{i}$. We define this as:

$$\vec{i} \in \mathbb{R}^{D_i}$$

where:

$$D_i = D_y \times N_y$$

and D_y is the dimensionality of the vector $\vec{y}$ and N_y is the number of steps. Multiplying the two give the dimensionality, D_i.

D. Extracting Behaviors From Feature Vectors

Once feature vectors $\vec{h}$ are extracted from data, behaviors can then be observed and extracted as well. The computation of a trajectory color map (Section III-D1), which is useful for human-readability, can also be used to cluster driving behaviors.

1) Dimension Reduction: Visualization is an important tool in understanding and presenting data. We used the technique proposed in [29] where a color map was used to represent driving features. With dimension reduction techniques, we can reduce the dimensionality of features, though even low dimensions can prove difficult to present in a 2D image or graph. One way of addressing this is to convert a feature vector to a color vector. These vectors consist of RGB (red, green, blue) values and can be mapped directly from any normalized 3-dimensional vector, since they have an equal number of components, by Eq. (3) where ϕ is a feature from the feature vector $\vec{h}$.

$$r\vec{g}b_{\phi} = \frac{h_{\phi} - h_{\phi}min}{h_{\phi}max - h_{\phi}min} \tag{3}$$

With the encoding process, of Eq. (2), we get feature vector $\vec{h}$ which is of low dimensionality. We can set the target dimensionality to 3, and assign them to red, green or blue values (Fig. 3). We can then relate this value back to its physical location in the real world, referring to the associated localized vehicle geolocation data. This generates a specific color for each point in space where the vehicle has traveled. Each route fed through the encoder section then generates a color map. The color map can be overlaid on a vector map containing topographical information, and we can see how the feature vector changes over the course of the trajectory (such as can be seen in Fig. 8 in the Section IV-B). At this stage we begin to see patterns unaided, the color map technique helping visualize the relationship between environment and driving behavior.

2) Clustering and Model Fitting: For clustering, we grouped together the values of the feature vector. Similar colors indicate trends in the original autoencoder input data. Clustering helps identify overlying driving behaviors instead of just driving actions. We tried different clustering techniques, including spectral clustering, KMeans, HDBScan, agglomerative clustering and mean shift. Compared to the other methods, spectral clustering displayed more defined and distinct clusters, and proved advantageous in our case due to the algorithm assuming a globular cluster distribution.

The various clusters are aggregated and used to fit several functions which represent the change in velocity over time and distance through the cluster. The result is a velocity profile for each cluster that can be later used in an autonomous agent. The functions used for the fitting process are either linear, logarithmic or polynomial; chosen as appropriate for the shape of the data in each cluster.

E. Trajectory Comparison

This section presents our method for comparing two trajectories. Trajectories are composed of different attributes that vary over a span of time. Even when the starting point and destination goal are the same, there can be many differences between two separate trajectories.

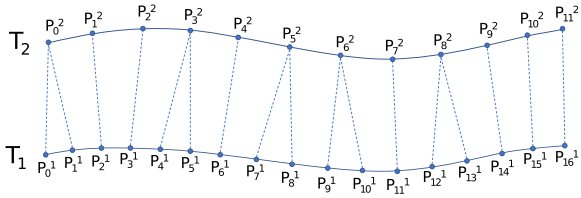


Fig. 4. To match trajectories T_1 and T_2 every point p_n^1 from T_1 is matched to the Euclidean distance closest point p_n^2 in T_2.

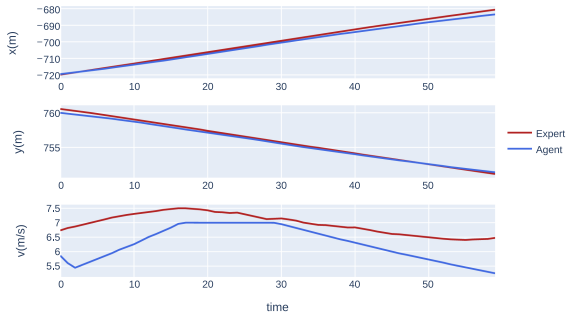


Fig. 5. A sample segment of two trajectories collected from our data, showing how the data compares in position and velocity.

A trajectory T is the data representation of a path that an entity has taken to get from one destination to another. The data is a series of time-ordered successive n geographical coordinates along with their state (Eq. (4)). While movement is a continuous activity, we sample the trajectory at a small enough time interval ΔT of 10 ms to give an adequate representation of this activity. In Eq. (5) we use position x, y, z, velocity v, acceleration acc and jerk je when defining trajectories.

$$T = (\vec{p_0} + \vec{p_1} + \cdots + \vec{p_n}) \tag{4}$$

$$\vec{p_n} = (x_n, y_n, z_n, v_n, acc_n, je_n) \tag{5}$$

In order to compare trajectories, we select attributes which, if changed, would indicate a diverging trajectory. To select these attributes, we performed a correlation analysis on several trajectories, to see how they would influence each other. When comparing two trajectories, we take every point from the first trajectory and find the closest point (determined by Euclidean Distance) from the second trajectory (Fig. 4). We match up every point in this way and then compare their values.

These attributes can each be used in an "attribute comparison score" Eq. (6), where a represents the attribute in question (Fig. 5). The score is a normalized mean difference between the points of two trajectories, with a score closer to 0 indicating similarity. The normalizer a_{norm} is chosen differently, depending on the attribute.

$$S_a(T_1, T_2) = \frac{\bar{a}_{diff}}{a_{norm}} \tag{6}$$

To get a quantitative comparison, considering multiple variables, we built a weighted score Eq. (7) to represent a directional

comparison score between two trajectories. The comparison is directional, since comparing T_1 to T_2 vs comparing T_2 to T_1 can have different results due to how matching works as shown in Fig. 4. Since trajectories can be of different lengths and a different number of points, the direction may affect the results. We take the normalized score of each attribute and multiply it by a weight. The sum of the normalized weights adds up to 1 (Eq. (8)). The sum of these scores gives a weighted average score, or "trajectory score," showing how similar the trajectories are, while a score of 0 indicates they are identical and higher scores indicates increasing divergence.

$$S(T_1, T_2) = \sum_{a=0}^{m} \omega_a S_a(T_1, T_2) \tag{7}$$

where,

$$\sum_{a=0}^{m} \omega_a = 1 \tag{8}$$

In the equation, the comparison attributes consist of distance, velocity, acceleration, and jerk. ω_a is the weight given to an attribute. The normalizer value in the single attribute equation used for normalization is determined specifically for each attribute. For distance, the maximum value is set to the average lane width, 3.5 m, of a typical road in Japan [41]. Realistically, if the trajectories are any further apart, they are not driving in the same lane. For velocity, acceleration and jerk we use a local normalizer, meaning we take the maximum value exhibited by the target trajectory.

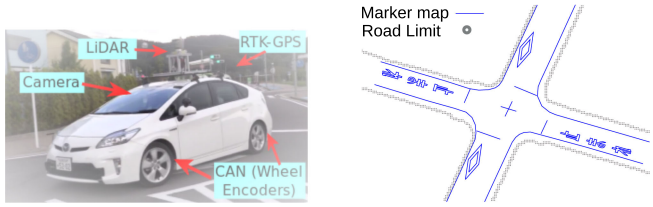
With this comparison method we can provide a relevant evaluation of our simulator-extracted trajectories to those of the driver data or between other trajectories.

IV. Experimental Procedure

This section explains the procedure for data collection and the preparation of the data to be used in the autoencoder. With the deep autoencoder we extracted feature vectors correlating to driving behavior. This behavior was then used to create a color map and behavior clusters. They were then used to fit velocity functions to be used as our driving models. The models were used as a basis for an AI agent used in a ROS environment.

A. Data Collection

For data collection, we recorded 5 expert drivers navigating through a residential neighborhood, providing over 300 tracks of data [8]. The drivers were given a route and instructions to drive in their own style, one that felt comfortable for them. As the drivers drove the vehicle, we collected data from a 360^o LiDAR (Velodyne $HDL-64E$), RTK-GPS with accuracy in open sky of ~ 0.1 m; and vehicle CAN-Bus data, at a sampling rate of 10 ms, providing the vehicle's speed, acceleration, brake pedal pressure, accelerator pedal position and steering angle (Fig. 6a and b and Table I). We had direct access to the CAN bus of the vehicle, provided to us by the manufacturer. The vector maps were created by a third-party company using their own Mobile Mapping System (MMS). We used our collected LiDAR data



(a) (b)

Fig. 6. Experimental vehicle and vector map. (a) Experimental vehicle equipped with a 360 LiDAR, RTK-GPS, CAN-Bus and monocular camera. (b) High definition vector map used to visualize and understand different driver behaviors.

TABLE I
Variable Details

Name	Description	Unit
x_n	x location component	m
y_n	y location component	m
z_n	z location component	m
v_n	velocity	m/s
acc_n	acceleration	m/s^2
br_n	brake position	$\%$
s_n	steering angle	rad

together with GPS and odometry to compute vehicle localization towards the 3D point cloud maps.

As a point of reference for comparisons done in Section V-A, we also used data collected from 4 elderly drivers without any professional driving experience [42]. They drove along the same trajectories as expert drivers.

First, the raw data is extracted and formatted into a sliding window vector; several points connected in time along the trajectory are combined as a single vector encapsulating previous vehicle state. For the input vector we take *speed*, *acceleration*, *brake*, and *steering angle* from the CAN data. We take the 4 input dimensions multiplied by 100-time steps (10 ms), creating a 400 dimension vector $\vec{i}$, as described in Section III-C.

B. Autoencoder Feature Extraction

The starting input layer of the network consists of 400 nodes, matching the dimensionality of our vector $\vec{i}$ and each dense layer of the encoder gradually reduces the number of nodes (400-300-150-64-16-3) and then is mirrored by the decoder with an equal number of layers containing equal node amounts, gradually increasing the number of nodes back to 400.

Fig. 7 shows the raw feature vector data plotting on it's 3-dimensional axes, while Fig. 8 shows the colors as applied to a geographical location according to an original sample route. Even by simply looking at the color patterns, one can appreciate which intersections require stops and which do not. This shows the potential to extract driving behaviors using this method.

For our network we chose to use "Adam" as the adaptive learning rate method due to its popularity for working well in practice and comparing favorably to other methods [43], [44]. In addition to using an autoencoder for feature extraction, we attempted to use principal component analysis (PCA) to create

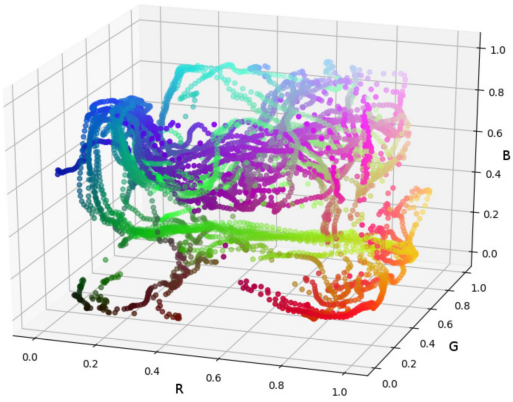


Fig. 7. The raw feature vector in three dimensions, showing the distribution of behavior from the encoded driving data in RGB.

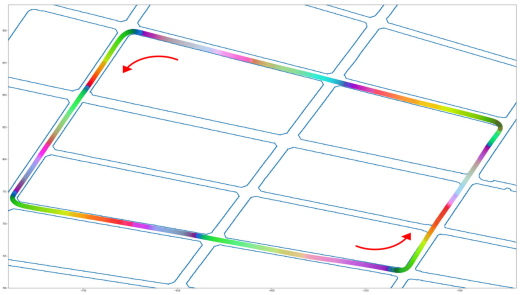


Fig. 8. Encoded expert driver data. The coloration shows various features of the vehicle. It can be seen how there are different behaviors depending on the driver's location on the road.

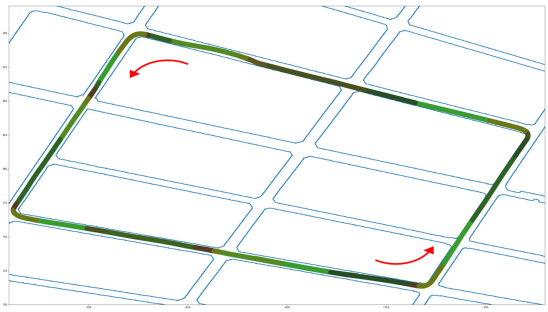


Fig. 9. Encoded expert driver data. This time using PCA instead of an autoencoder. The resulting color map is not as easily discernible or separable as the autoencoder results.

the same behavior color map. However, as can be seen in Fig. 9, the behaviors are not captured as well as by the autoencoder. The darker, more similar colors, show that it was not as effective at separating behaviors. This may be due to autoencoders using non-linear activation functions, whereas PCA is restricted to a linear map. A single layer autoencoder is approximately equivalent to PCA. In the case of a deep autoencoder, the multiple layers add flexibility to the learning process.

C. Clustering

A selected k value of 9 with spectral clustering, using a k-nearest neighbors connectivity matrix, results in our nine unique clusters. This represents driving behaviors, as shown in Fig. 10

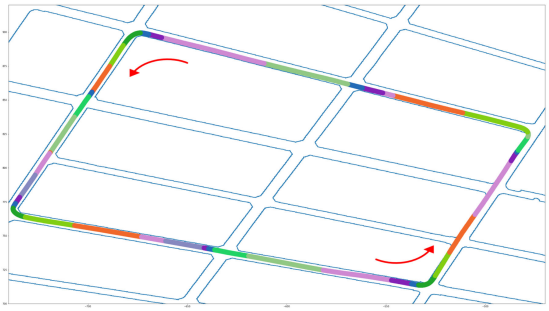


Fig. 10. Clustering results: Each colored segment/cluster represents a driving behavior sharing characteristics with segments of the same color. The red arrow indicates a counterclockwise driving direction.

TABLE II
CLUSTERING RESULTS

ID	color	swatch	behavior state	eq.
0	Pink		maintain speed, decreasing acceleration	(11)
1	Blue		gentle acceleration from stop	(10)
2	Purple		sharp deceleration to stop	(9)
3	Orange		deceleration from high speed	(9)
4	Pear		decreasing acceleration	(11)
5	Green		acceleration around left turn	(9)
6	Lime		steady acceleration to high speed	(9)
7	Emerald		sharp acceleration	(12)
8	Steel		sharp deceleration from high speed	(9)

and Table II. Clustering based on the feature vector gives clumps of similar color-coded behavior. These groupings of behavior could then be used as a basis for velocity profiling.

Please refer to Fig. 10 and II where we further defined a possible driving behavior for each color. Pink segments Insert Graphicare seen in long straight sections of the map and are associated with maintaining speed alongside gentle acceleration and deceleration at the beginning and end of the cluster respectively. The Blue Insert Graphic cluster is only seen at the beginning of intersections and indicates a slow acceleration from a zero or low velocity, as the driver slowly lets off the brake. Purple Insert Graphic immediately precedes blue clusters and indicates the vehicle is approaching a stop. Orange Insert Graphic, often precedes a pink cluster and indicates a deceleration from a high speed. The various shades of green indicate varying states of acceleration. Pear Insert Graphic shows a gentle curved acceleration at high speed, Green Insert Graphic shows acceleration after making a left turn, Lime Insert Graphic indicates steady acceleration up to a high speed, and Emerald Insert Graphic shows a sharp curved acceleration from low speed all the way up to high speed. Finally, Steel Insert Graphic was the start of a sharp deceleration from high speed.

D. Velocity Fitting

Clustered feature vectors, and the previous observation of behaviors, give a basis to build a velocity model. We have several examples of movement through a given section of road which can be approximated with a specific model. The simplest way to do this is with a function fit using the non-linear squares

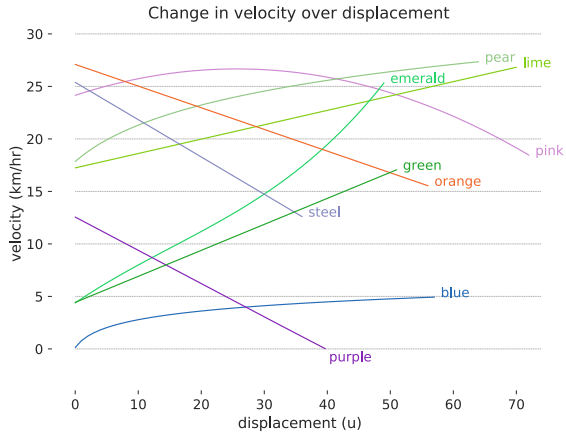


Fig. 11. Velocity fitting results: Each curve in this figure shows the results of fitting a function to each cluster. These functions fit the cluster data with a certain amount of regression error, therefore, we chose the ones minimizing that error for each cluster.

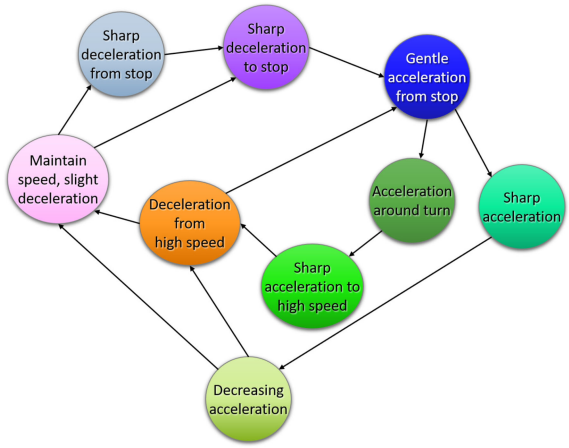


Fig. 12. The driving behavior transition diagram. Each circle shows a different acceleration/deceleration behavior (with corresponding colors and behaviors as mentioned in Section IV-C and Table II). The transitions occur based on geographical location coordinates (as shown in Fig. 10).

TABLE III
FIT PARAMETERS

Cluster	Fit Type	Parameters (d, e, j, k)
Pink	Quadratic	-0.004, 0.196, 24.152
Blue	Logarithmic	1.326, 1.567, 2.585, -1.724
Purple	Linear	0.137, 17.240
Orange	Cubic	$1.30e^{-4}$, -0.006, 0.407, 4.385
Pear	Logarithmic	4.407, 8.384, 1.705, 6.126
Green	Linear	-0.355, 25.386
Lime	Linear	-0.206, 27.096
Emerald	Linear	-0.317, 12.559
Steel	Linear	0.248, 4.433

Table III shows the results for the parameters d, e, j and k (where applicable) for each fit velocity profile per cluster.

method. Using our data, we can fit function parameters to match. In the case of driving data, we chose several equations that match the basic patterns of acceleration and deceleration through each cluster. The nine clusters were individually fit using one of four different functions:

$$V_1(x) = dx + e \tag{9}$$

$$V_2(x) = d \log e(x + j) + k \tag{10}$$

$$V_3(x) = dx^2 + ex + j \tag{11}$$

$$V_4(x) = dx^3 + ex^2 + jx + k \tag{12}$$

The velocity equations V_i used are chosen from linear Eq. (9), logarithmic Eq. (10), quadratic Eq. (11) and cubic Eq. (12) functions. The variables d, e, j and k are the function parameters, modified and fit by the fitting function; while the variable x is the input displacement. The resulting fits are shown in Fig. 11, and the fit parameters in Table III.

E. Behavior Transition Diagram

The clustered behaviors and the observed relationships allow us to construct a behavior transition diagram which can then be used to create an autonomous driving agent. As presented in Section IV-F, in the ROS simulator environment, the agent follows a path generated by a global planner around the reconstructed suburban neighborhood, where our original data came from. As the agent travels through, it uses the corresponding velocity fit (Fig. 11) according to the behavior assigned to that location (see Fig. 10). The velocity and position of the agent is recorded and compared to the original driver moving through that same route in Section V-A.

Fig. 12 shows the behavior transition diagram representing the flow and relative position of each driving behavior, as described in Section IV-C. Each transition is based on the physical geographical location of the vehicle. Each behavior has one or more source and connecting behaviors. For example, the pink behavior follows from either the orange or pear behavior, and is followed by either the steel or purple behavior.

The task to extract useful information or behaviors from data in a supervised manner is a hard task. Our approach makes use of deep learning methods to extract useful features in an unsupervised way from the driving data. With this feature extraction process we create behavior models for our driving agent to drive autonomously.

F. Driving Agent

The agent created using the driving behavior transition diagram was tested in a simulator.[2] Due to safety concerns and regulations, it is easier to test new autonomous driving agents in a simulation as opposed to the real world. The simulated agent could theoretically run in a real environment with similar performance to that of the simulation.

The cluster following behavior of our agent is created as a layer over an existing agent simulation platform. The base agent is able to receive a global path trajectory and then follow it using real world physics and parameters [8]. Our driving behavior

[2]A video showing the simulator in action can be seen here: [Online]. Available: https://youtu.be/y5GZPUl5w2Q

cluster following layer will then modulate the velocity based on the appropriate behavior selected from the behavior transition diagram. This entire system is implemented in ROS (Robot Operating System).

The behavior transition flow is presented with the following equations:

$$\delta(\rho, c_n) = \min \sqrt{((c_{n_x} - \rho_x)^2 (c_{n_y} - \rho_y)^2)} \tag{13}$$

First, the agent's position $\delta(\rho, c_n)$ and the closest behavior cluster c_n is found based on geographical coordinates ρ; determined through the vector map and vehicle localization.

$$C_{pos} = |(len(c_n) - \delta(\rho, c_n))/len(c_n)| \tag{14}$$

The position within the cluster C_{pos}, or how far into the cluster the vehicle has driven, is determined.

$$v_{c_n} = V(C_{pos}) \tag{15}$$

The position is fed into the cluster's corresponding velocity profile V to determine the desired velocity v from the cluster c_n.

The simulated environment itself is a replication of the same rural Japanese neighborhood where our driving data was gathered from. The various road features such as lanes, stop lines, and static obstacles are present.

With our driving behavior cluster following layer giving commands to the general agent, we record the performance and driving data of the virtual vehicle as if it were a real vehicle; tracking velocity and position. This trajectory data can then be used to compare the results of our agent to that of real drivers. The agent attempts to match the given driving instructions. In the event that something interrupts this operation, such as an emergency brake, the agent will accelerate back to the suggested velocity once it regains control (as given by the current behavior). When operating the driving agent in the simulator environment, the agent can be placed at any point near the desired path and it will attempt to match the velocity required by the appropriate behavior in that area.

V. Results

This section presents the results of our driving agent experiments, comparing its recorded trajectory against those of expert and elderly drivers.

A. Trajectory Comparison Results

Qualitatively, we can observe the differences between trajectories in Fig. 13. Compared to the similar figure in [1], we added smoother transition logic between behavior clusters in our driving agent. We did this by blending the velocity near the beginning/end of each behavior with the final/initial velocity of the previous/next behavior. This created a more similar overall appearance between trajectories, as well as a better trajectory comparison score.

Quantitatively, using the scoring method from Section III-E, we illustrate the comparison between expert drivers and our modeled driving agents.

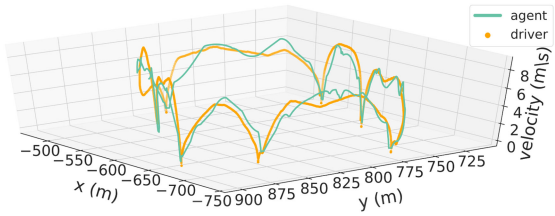


Fig. 13. Visualization of trajectory difference between real driver data (in yellow) and the simulated agent (in teal). The agent's trajectory appears to behave closely to the expert. (See Fig. 10).

TABLE IV
EXPERT DRIVER 2 TO EXPERT DRIVER 1

name	score	mean	median	std	max
distance	0.113	0.396	0.284	0.356	1.756
velocity	0.066	0.578	0.481	0.484	3.481
acceleration	0.136	0.458	0.329	0.505	4.787
jerk	0.046	1.611	0.988	3.114	42.069
average	0.090				

TABLE V
EXPERT DRIVER 3 TO EXPERT DRIVER 1

name	score	mean	median	std	max
distance	0.079	0.278	0.196	0.372	2.683
velocity	0.063	0.552	0.426	0.543	5.059
acceleration	0.112	0.378	0.197	0.507	5.305
jerk	0.047	1.643	0.907	3.438	45.714
average	0.075				

Tables IV, and V Comparisons between trajectory pairs of the expert drivers themselves.

TABLE VI
AI AGENT TO EXPERT DRIVER

name	score	mean	median	std	max
distance	0.093	0.326	0.298	0.234	1.758
velocity	0.111	0.972	0.831	0.735	4.032
acceleration	0.213	0.72	0.535	0.635	4.621
jerk	0.069	2.434	1.281	3.932	34.072
average	0.122				

To show the meaningfulness of these comparisons, we show in Tables IV and V comparisons between expert drivers, showing that the comparison score is significantly lower than both elderly drivers and the agent (as seen in later comparisons, Tables VI to X), indicating similarity between the experts' driving styles.

In Table VI we compared the performance of the driving agent to the expert driver. Note that the closer to 0 a score is, the more alike it is to the target trajectory, while a score closer to 1 or above is indicative of completely different behavior. This shows that our agent has a trajectory that is close to the expert in terms of distance, velocity, and jerk and also quite similar in terms of acceleration. With an average score of 0.122, we can say that the trajectory is similar to that of the expert driver.

Without a point of reference, it is hard to compare how similar a trajectory actually is. To provide such a reference, we compare the performance of a selection of elderly drivers in Tables VII to

TABLE VII
ELDERLY DRIVER 1 TO EXPERT DRIVER

name	score	mean	median	std	max
distance	0.154	0.538	0.495	0.309	4.642
velocity	0.228	2.903	1.814	2.463	7.096
acceleration	0.18	0.789	0.722	0.58	5.583
jerk	0.09	3.961	2.346	4.958	48.993
average	0.163				

TABLE VIII
ELDERLY DRIVER 2 TO EXPERT DRIVER

name	score	mean	median	std	max
distance	0.153	0.536	0.475	0.34	3.148
velocity	0.572	7.294	6.417	4.693	12.722
acceleration	0.15	0.656	0.528	0.543	5.889
jerk	0.077	3.382	2.5	4.047	48.364
average	0.238				

TABLE IX
ELDERLY DRIVER 3 TO EXPERT DRIVER

name	score	mean	median	std	max
distance	0.127	0.444	0.406	0.412	3.976
velocity	0.27	3.444	1.231	4.57	12.678
acceleration	0.116	0.507	0.298	0.538	5.424
jerk	0.08	3.489	1.926	5.244	45.278
average	0.148				

TABLE X
ELDERLY DRIVER 4 TO EXPERT DRIVER

name	score	mean	median	std	max
distance	0.128	0.448	0.407	0.382	2.598
velocity	0.158	2.008	1.239	1.696	4.267
acceleration	0.163	0.717	0.639	0.55	5.71
jerk	0.062	2.725	1.111	4.708	46.389
average	0.128				
average elderly driver score	0.169				

Table VI to X. Comparisons between trajectory pairs as described in Section III-E. The *score* is the value as described in that section and the *average* is evenly weighted between all attributes. The *mean* column shows the mean difference in each attribute at each point in time across the whole trajectory. The subsequent trajectories are the same but indicate the *median*, standard deviation (*std*) and *max* values respectively.

X to that of an expert driver. The results show that, on average, our agent performs more closely to that of an expert driver than the elderly drivers, and the elderly drivers having only exceeded the agent with scores in acceleration by a small margin. Importantly, we see that velocity similarity is worse for the elderly drivers. We can also see in Fig. 14, where the trajectory of an elderly driver is compared to the expert, on the right side of the figure, on the middle arch in velocity, that the elderly driver does not slow down like the expert does, when moving through an unsignaled intersection. The agent, on the other hand, does slow down, similar to the expert. This demonstrates the agent was able to accurately model the experienced driver, allowing

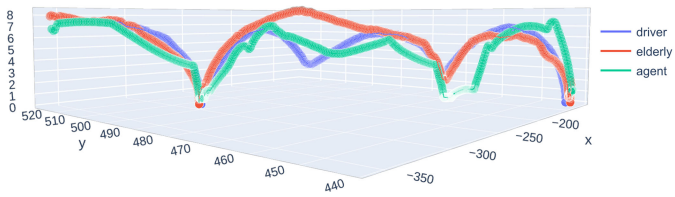


Fig. 14. Visualization of trajectory difference between expert driver in blue, elderly driver in red and our driving agent in green.

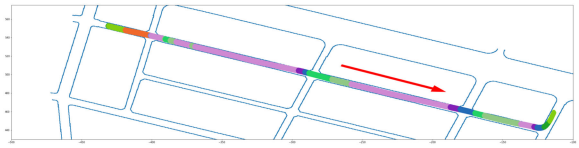


Fig. 15. Encoded behaviors of alternate route. Note the frequent behavior change near the beginning of the trajectory, this may indicate missing behaviors in the original training behavior.

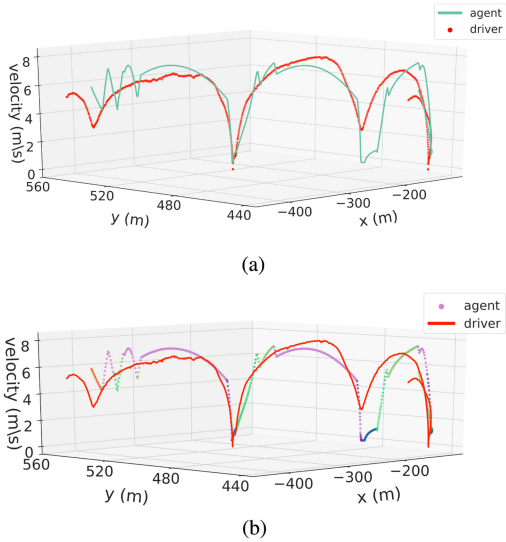


Fig. 16. Visualization of trajectory differences on an alternate route. Perturbations due to frequently changing behaviors, as can be seen in the lower figure, are likely caused by new behaviors in the new data, not present in the original training set. (a) Driving agent/expert driver comparison. (b) Driving agent/expert driver comparison, with encoded behaviors.

it to drive proactively, thus reducing the risk from yet unseen entities approaching a blind intersection. We can say that the closer the score of the agent is to the expert driver, the more proactive the agent is.

Notably, as can be seen in the "max" deviation (i.e. the maximum amount the value differed from that of the target trajectory) columns of the agent and elderly driver vs expert driver comparisons, that the values are significantly lower for the driving agent.

B. Alternate Route Comparison

In Fig. 13 we see the data used are trajectories that have the same path as the trajectories used for training the autoencoder. In

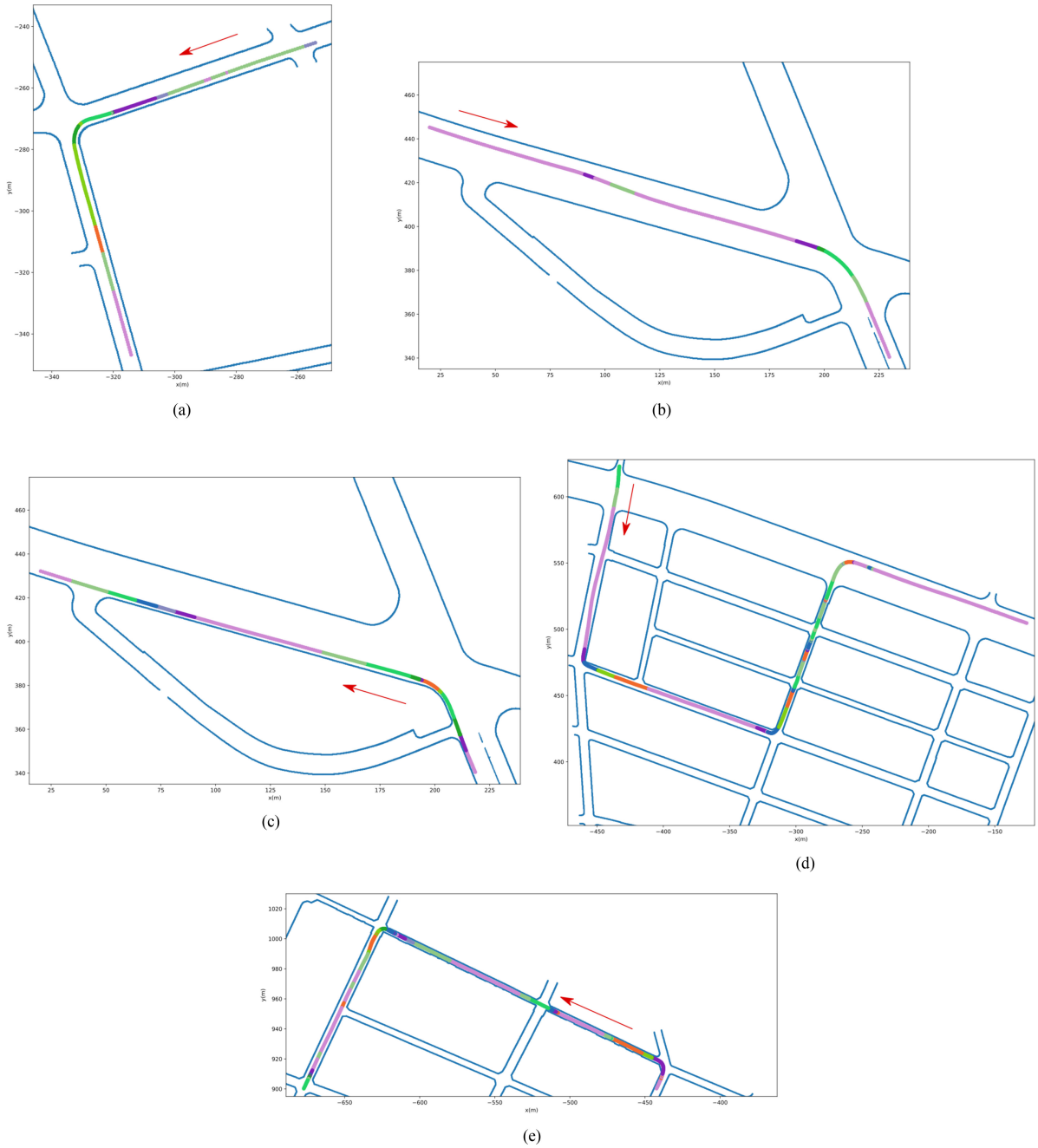


Fig. 17. Five additional environments in which the agent navigated through. Each environment examined was not from the original training dataset. The trajectories cover additional new scenarios, such as a lane change or right turns, and still have trajectory scores indicating similarity. (a) A trajectory with a left turn and passing through a T-intersection. (b) A lane change preceding a right turn. (c) The same area as Fig. 17b, but traversing in the opposite direction. (d) A trajectory through several intersections, left and right turns, and a multi-laned road. (e) Another trajectory showing various stopping behaviors at each intersection.

Fig. 16a however, we use a trajectory completely unassociated with the original training data set. We used the encoder trained in Section III-C to encode the trajectory which resulted in Fig. 15. This new color map was then used by the agent to attempt to drive the new path.

Like in Fig. 13, Fig. 16a shows that the agent attempts to follow the general behavior of the expert driver, however in this case, there are some areas that seem to have a lot of noise. We can see near the beginning of the trajectory in Fig. 16b that the behavior changes frequently before settling into a single behavior. This is likely caused by the new data requiring behaviors that were not learned in the initial training dataset.

There are several other new areas in addition to the one mentioned above that were also tested. The results can be seen in Fig. 17 and Tables XI to XV where 5 additional environments are presented. The agent drove in each area, and was compared to an expert driver. In Fig. 17b and c we see the same road in both trajectories, but traversing in opposite directions, showing how the behavior is dependent not just on the location, but on the context of the situation as well.

TABLE XI
AI AGENT TO EXPERT DRIVER FIG. 17A

name	score	mean	median	std	max
distance	0.056	0.197	0.114	0.255	1.234
velocity	0.194	1.701	1.421	1.151	3.639
acceleration	0.462	1.364	1.482	0.888	4.29
jerk	0.123	3.638	1.582	6.113	40.727
average	0.209				

TABLE XII
AI AGENT TO EXPERT DRIVER FIG. 17B

name	score	mean	median	std	max
distance	0.06	0.21	0.2	0.121	0.555
velocity	0.17	1.982	1.816	1.235	4.633
acceleration	0.483	1.173	0.852	1.021	4.615
jerk	0.112	4.32	1.791	6.905	47.389
average	0.206				

TABLE XIII
AI AGENT TO EXPERT DRIVER FIG. 17C

name	score	mean	median	std	max
distance	0.045	0.156	0.127	0.123	1.172
velocity	0.211	2.458	2.371	1.609	8.288
acceleration	0.383	1.272	1.071	0.988	4.537
jerk	0.094	3.127	1.523	4.793	43.9
average	0.183				

TABLE XIV
AI AGENT TO EXPERT DRIVER FIG. 17D

name	score	mean	median	std	max
distance	0.109	0.381	0.228	0.348	1.253
velocity	0.128	1.791	1.681	1.395	6.714
acceleration	0.291	1.211	0.77	1.124	4.84
jerk	0.112	4.671	1.826	7.071	48.255
average	0.16				

TABLE XV
AI AGENT TO EXPERT DRIVER FIG. 17E

name	score	mean	median	std	max
distance	0.045	0.157	0.111	0.16	0.944
velocity	0.239	1.55	1.47	1.132	6.144
acceleration	0.297	1.159	0.735	1.099	4.695
jerk	0.122	4.936	2.267	7.366	58.424
average	0.176				

Comparisons between trajectory pairs of the Agent and expert drivers in various areas (Fig. 17) not in the original training data scope.

C. Other Expert Driver Comparisons

We also compared our agent's results to the trajectories of other expert drivers going through the same route. The results can be seen in Table XVI and XVII. Even though the agent was trained on a different expert driver's data, the scores are still similar. This again shows that the agent is driving more akin to drivers known to drive proactively.

TABLE XVI
AI AGENT TO EXPERT DRIVER 2

name	score	mean	median	std	max
distance	0.106	0.373	0.285	0.296	1.879
velocity	0.131	1.216	0.996	0.96	4.514
acceleration	0.225	0.848	0.662	0.721	5.042
jerk	0.056	2.419	1.284	3.865	34.686
average	0.130				

TABLE XVII
AI AGENT TO EXPERT DRIVER 3

name	score	mean	median	std	max
distance	0.094	0.329	0.294	0.207	1.763
velocity	0.149	1.34	1.032	1.143	5.346
acceleration	0.207	0.816	0.621	0.694	4.473
jerk	0.063	2.432	1.253	3.932	36.868
average	0.128				

Tables XVI and XVII Comparisons between trajectory pairs of the Agent and other expert drivers.

VI. DISCUSSION

One of the main reasons we don't yet have fully autonomous cars on the road today is the lack of high-level cognitive function in driving agents. This research shows that it is possible to learn a subset of these functions and therefore emulate human-like driving behavior. While one of the main limitations in this work is that it does not consider dynamic obstacles such as traffic or pedestrians while navigating, it shows that we can learn how to drive in the absence of obstacles, which is important to achieve safe and comfortable navigation.

The results shown in this work are specific to Japan, but still shows that given the data, an agent can learn how to drive in a human-like fashion.

We chose to define a new method for trajectory comparison, instead of using the modified Hausdorff distance [45], so that we could have a better overall comparison of each point in the trajectory as well as consider multiple attributes.

The trajectory score used in this work shows that our agent can drive better than elderly drivers in terms of velocity and jerk, as shown in Fig. 14.

If we were to model driving behavior directly, we might create a state machine based on theory in works such as [46], [47]. We would have several states: acceleration, deceleration, turning, stopping and velocity keeping. This kind of state machine can be used to manually create an agent with velocity models for each state. This would be an example of a handpicked feature driven approach to model driving. In the case of our feature extraction method using deep learning, we have instead extracted behaviors encompassing several of these states at any given time. An example behavior, such as a "sharp left turn," might include acceleration, deceleration and velocity keeping all in one. This makes it difficult to create a pure state machine from these behaviors. Instead, we created a behavior transition diagram Fig. 12. This diagram shows the behaviors in relation to one another and how they follow each other over the course of a trajectory. The transitions from behavior to behavior are map

location based, as can be seen in the associated color map created from the driving features.

Another benefit of our technique is that we can easily classify segments of a trajectory, consisting of different actions, as a single behavior, such as a "left turn" or "crossing an intersection". The benefit of our data-driven approach over the model based one is being able to extract the features directly from an actual driver. We know they are valid in the sense that they come from an established expert driver able to perform complex driving tasks in a real-life environment.

If we look at very recent work in driving behavior extraction and analysis we see that a common trend is to use Deep Reinforcement Learning (DRL) [48], [49]. Some recent papers are also focusing on Attention-based DRL [50], [51]. These techniques are likely to be used more often in the future, and will be a logical next step to investigate for this work as well.

VII. Conclusion

In this work, we examined the process of extracting driving behaviors from expert drivers. We used a deep autoencoder network to process large volumes of data to extract latent features. We clustered these latent features into behaviors and created velocity profiles. This allowed us to create an autonomous agent to correctly select the proper behavior to use depending on the environment it was in. Our experimental results, including our trajectory scoring system, show comparable results to that of expert drivers in the same environment. We were also able to demonstrate that these extracted behaviors are applicable to other similar environments. We show that our agent was able to use the professional experience, gained from years of human driving instructing, in order to drive proactively in several urban environments (including those of the training set as well as those unknown).

Future development stemming from this work would have the goal to handle dynamic obstacles. We would like to use our techniques to extract behaviors and create driving agents that are able to navigate while avoiding dynamic obstacles, such as traffic and pedestrians, like in the work by Bansal *et al.* [36]. Additional data sources like LiDAR and camera could be used to accomplish this. However, it would require ground removal techniques for LiDAR and semantic segmentation for camera data. New neural network designs for training and machine learning purposes.

Our trajectory scoring technique could also be used for other applications, such as trajectory matching, either for localization or sensor calibration. For example, if you have multiple sensors in a room tracking a person's movement, this comparison method could be used to determine the similarity of each sensor's tracked trajectory. Then the sensors' alignment can be configured by minimizing the score.

The end-goal of this technique would be to generate required driving behaviors for an autonomous agent simply by giving it a path and a vector map. This can likely be done with extensive data to train more types of behaviors and a recurrent neural network (RNN), such as a long term-short term memory (LSTM) network to identify the required behaviors along the path. There is potential in using "Cycle Variational AutoEncoders" (Cycle-VAE) and other "Sequence to Sequence" (S2S) networks for use in extracting features from continuous datasets, such as driving data.

References

[1] K. Sama, Y. Morales, N. Akai, H. Liu, E. Takeuchi, and K. Takeda, "Driving feature extraction and behavior classification using an autoencoder to reproduce the velocity styles of experts," in *Proc. 21st Int. Conf. Intell. Transp. Syst.*, Nov. 2018, pp. 1337–1343.

[2] H. Ball, "The 5 breakthroughs powering the self-driving car boom," *INC. Magazine*, New York, NY, USA, Feb. 2017.

[3] S. Thrun *et al.*, "Stanley: The robot that won the DARPA grand challenge," *J. Field Robot.*, vol. 23, no. 1, pp. 661–692, Jun. 2006.

[4] C. Urmson *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *J. Field Robot.*, vol. 25, no. 1, pp. 425–466, Jun. 2008.

[5] M. Elbanhawi, M. Simic, and R. Jazar, "In the passenger seat: Investigating ride comfort measures in autonomous cars," *IEEE Intell. Transp. Syst. Mag.*, vol. 7, no. 3, pp. 4–17, 2015.

[6] T. Ram and K. Chand, "Effect of drivers' risk perception and perception of driving tasks on road safety attitude," *Transp. Res. F, Traffic Psychol. Behav.*, vol. 42, pp. 162–176, 2016.

[7] J. Bärgman, K. Smith, and J. Werneke, "Quantifying drivers' comfort-zone and dread-zone boundaries in left turn across path/opposite direction (LTAP/OD) scenarios," *Transp. Res. F, Traffic Psychol. Behav.*, vol. 35, pp. 170–184, 2015.

[8] Y. Yoshihara, Y. Morales, N. Akai, E. Takeuchi, and Y. Ninomiya, "Autonomous predictive driving for blind intersections," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2017, pp. 3452–3459.

[9] Y. Morales, Y. Yoshihara, N. Akai, E. Takeuchi, and Y. Ninomiya, "Proactive driving modeling in blind intersections based on expert driver data," in *Proc. IEEE Intell. Vehicles Symp.*, 2017, pp. 901–907.

[10] T. Kumagai and A. Takahashi, "Influence of repeated experience on unsignalized intersection crossing behavior of drivers without right-of-way," in *Proc. Int. Conf. Human-Comput. Interact.*, Jul. 2013, pp. 323–326.

[11] National Highway Traffic Safety Administration (NHTSA), *Fatality Analysis Reporting System (FARS)*, 2010. [Online]. Available: www.nhtsa.gov/FARS

[12] M. F. Mitman and D. R. Ragland, "Driver/pedestrian understanding and behavior at marked and unmarked crosswalks," Jul. 2008. [Online]. Available: https://escholarship.org/uc/item/7xn8m790

[13] B. Bougler, D. Cody, and C. Nowakowski, *Crash Factors in Intersection-Related Crashes: An On-Scene Perspective*. Berkeley, CA, USA: California Partners for Advanced Transportation Technology, 2008.

[14] E. H. Choi and United States National Highway Traffic Safety Administration, Crash Factors in Intersection-Related Crashes: An On-Scene Perspective (Series NHTSA Technical Report DOT HS 811 366). Scotts Valley, CA, USA: Createspace Independent Pub, 2010. [Online]. Available: https://books.google.com/books?id=Pxc2nQAACAAJ&dq

[15] D. Greene *et al.*, "An efficient computational architecture for a collision early-warning system for vehicles, pedestrians, and bicyclists," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 942–953, Dec. 2011.

[16] Y. Morales *et al.*, "Visibility analysis for autonomous vehicle comfortable navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 2197–2202.

[17] Y. Morales, A. Watanabe, F. Ferreri, J. Even, K. Shinozawa, and N. Hagita, "Passenger discomfort map for autonomous navigation in a robotic wheelchair," *Robot. Auton. Syst.*, vol. 103, pp. 13–26, 2018.

[18] S. Han, S. Xu, W. Meng, and C. Li, "Dense-device-enabled cooperative networks for efficient and secure transmission," *IEEE Netw.*, vol. 32, no. 2, pp. 100–106, Mar./Apr. 2018.

[19] W. Liang, Z. Li, H. Zhang, S. Wang, and R. Bie, "Vehicular ad hoc networks: Architectures, research issues, methodologies, challenges, and trends," *Int. J. Distrib. Sensor Netw.*, vol. 11, no. 8, 2015, Art. no. 745303.

[20] R. Kenntner-Mabiala, Y. Kaussner, S. Hoffmann, and M. Volk, "Driving performance of elderly drivers in comparison to middle-aged drivers during a representative, standardized driving test in real traffic," *ZVS*, vol. 62, pp. 73–76, 2016.

[21] G. McGwin, Jr., and D. B. Brown, "Characteristics of traffic crashes among young, middle-aged, and older drivers," *Accident Anal. Prevention*, vol. 31, no. 3, pp. 181–198, May 1999.

[22] D. F. Preusser, A. F. Williams, S. A. Ferguson, R. G. Ulmer, and H. B. Weinstein, "Fatal crash risk for older drivers at intersections," *Accident Anal. Prevention*, vol. 30, no. 2, pp. 151–159, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0001457597000900
[23] K. Sama, Y. Morales, N. Akai, E. Takeuchi, and K. Takeda, "Retrieving a driving model based on clustered intersection data," in *Proc. 3rd Int. Conf. Control Robot. Eng.*, Apr. 2018, pp. 222–226.
[24] K. Sama, Y. Morales, N. Akai, E. Takeuchi, and K. Takeda, "Learning how to drive in blind intersections from human data," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Miyazaki, Japan, Oct. 2018, pp. 317–324.
[25] S. Li *et al.*, "Driving feature extraction from high and low skilled drivers in curve sections based on machine learning," *J. Mech. Syst. Transp. Logistics*, vol. 6, no. 2, pp. 111–123, 2013.
[26] G. Gao and M. V. Wuthrich, *Feature Extraction from Telematics Car Driving Heatmaps*, Nov. 2017. [Online]. Available: https://papers.ssrn.com/abstract=3070069
[27] H. Al-Sahaf, M. Zhang, A. Al-Sahaf, and M. Johnston, "Keypoints detection and feature extraction: A dynamic genetic programming approach for evolving rotation-invariant texture image descriptors," *IEEE Trans. Evol. Comput.*, vol. 21, no. 6, pp. 825–844, Dec. 2017.
[28] H. Liu, T. Taniguchi, Y. Tanaka, K. Takenaka, and T. Bando, "Essential feature extraction of driving behavior using a deep learning method," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2015, pp. 1054–1060.
[29] H. Liu, T. Taniguchi, Y. Tanaka, K. Takenaka, and T. Bando, "Visualization of driving behavior based on hidden feature extraction by using deep learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 9, pp. 2477–2489, Sep. 2017.
[30] Y. Yang, Z. Wu, Q. Xu, and F. Yan, "Deep learning technique-based steering of autonomous car," *Int. J. Comput. Intell. Appl.*, vol. 17, no. 2, 2018, Art. no. 1850006.
[31] E. Yurtsever, C. Miyajima, and K. Takeda, "A traffic flow simulation framework for learning driver heterogeneity from naturalistic driving data using autoencoders," *Int. J. Autom. Eng.*, vol. 10, no. 1, pp. 86–93, 2019.
[32] M. Treiber and D. Helbing, "Explanation of observed features of self-organization in traffic flow," Jan. 1999. [Online]. Available: https://cds.cern.ch/record/377250
[33] F. Codevilla, M. Miiller, A. Lopez, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 4693–4700.
[34] M. Shimosaka, K. Nishi, J. Sato, and H. Kataoka, "Predicting driving behavior using inverse reinforcement learning with multiple reward functions towards environmental diversity," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2015, pp. 567–572.
[35] M. Kuderer, S. Gulati, and W. Burgard, "Learning driving styles for autonomous vehicles from demonstration," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 2641–2646.
[36] M. Bansal, A. Krizhevsky, and A. Ogale, "Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst," in *Proc. Robot., Sci. Syst.*, FreiburgimBreisgau, Germany, Jun. 2019, doi: 10.15607/RSS.2019.XV.031.
[37] E. Pearson, "Comfort and its measurement – A literature review," *Disability Rehabil. Assistive Technol.*, vol. 4, pp. 301–310, 2009.
[38] L. Svensson and J. Eriksson, "Tuning for ride quality in autonomous vehicle: Application to linear quadratic path planning algorithm," *Diss., Uppsala Univ., Division Syst. Control*, no. 15030, p. 61, 2015.
[39] Eclipse Deeplearning4j Development Team, "Deeplearning4j: Open-source distributed deep learning for the JVM." *Apache Softw. Found. License 2.0*, [Online]. Available: http://deeplearning4j.org, 2017.
[40] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[41] Road Bureau Ministry of Land, Infrastructure, Transport and Tourism, "2018 roads in japan," 2018. [Online]. Available: http://www.mlit.go.jp/road/road_e/pdf/ROAD2018web.pdf
[42] Y. Yoshihara, E. Takeuchi, and Y. Ninomiya, "Accurate analysis of expert and elderly driving at blind corners for proactive advanced driving assistance systems," in *Proc. 95th Annu. Meeting Transp. Res. Board*, Jan. 2016, pp. 16–1992.
[43] S. Ruder, "An overview of gradient descent optimization algorithms," *CoRR*, 2016. [Online]. Available: http://arxiv.org/abs/1609.04747
[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ICLR*, vol. abs/1412.6980, 2015.
[45] M. P. Dubuisson and A. K. Jain, "A modified hausdorff distance for object matching," in *Proc. 12th Int. Conf. Pattern Recognit.*, Oct. 1994, vol. 1, pp. 566–568.
[46] A. Kurt, "Hybrid-state system modelling for control, estimation and prediction in vehicular autonomy," *Electron. Thesis Diss.*, 2012. [Online]. Available: https://etd.ohiolink.edu/
[47] V. N. Gadepally, "Estimation of driver behavior for autonomous vehicle applications," Ohio State Univ., 2013. [Online]. Available: https://books.google.co.jp/books?id=QgpMnwEACAAJ
[48] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," 2019, *arXiv:1906.05113v2*.
[49] B. R. Kiran *et al.*, "Deep reinforcement learning for autonomous driving: A survey," 2020, *arXiv:2002.00444*.
[50] E. Aksoy, A. Yazici, and M. Kasap, "See, attend and brake: An attention-based saliency map prediction model for end-to-end driving," 2020, *arXiv:2002.11020*.
[51] X. Fu, X. Di, and Z. Mo, "When do drivers concentrate? Attention-based driver behavior modeling with deep reinforcement learning," *CoRR*, 2020. [Online]. Available: https://arxiv.org/abs/2002.11385

Kyle Sama received the B.Eng. in software engineering from Concordia University, Montreal, QC, Canada, in 2012. From 2017 to 2019, he was a Research Student with Nagoya University, Nagoya, Japan, where he is currently working toward the Ph.D. degree. His main research interests include deep learning for autonomous vehicles and extracting driving behavior from human data.

Yoichi Morales (Member, IEEE) received the M.Eng. and Ph.D. degrees in robotics in 2006 and 2009, respectively, from Intelligent Robot Laboratory, University of Tsukuba, Tsukuba, Japan, where he was a Postdoctoral Researcher for six months until 2009. He was a Researcher with the ATR Intelligent Robotics and Communication Laboratories in Kyoto, Kyoto, Japan, where he is currently a Collaborative Researcher. He is currently a designated Associate Professor with the University of Nagoya, Nagoya, Japan. His research interests include autonomous navigation, spatial cognition, perception, and environment modeling. He is a member of the IEEE Robotics and Automation Society and the Robotics Society of Japan.

Hailong Liu (Member, IEEE) received the B.E. degree in information science and engineering from Ritsumeikan University, Kyoto, Japan, in 2013, and the M.S. and Ph.D. degrees from the Graduate School of Information Science and Engineering of the same university in 2015 and 2018, respectively. Meanwhile, he had been a Research Fellow (DC2) of Japan Society for the Promotion of Science from April 2016 to March 2018. He is currently a Researcher with the Graduate School of Informatics, Nagoya University, Nagoya, Japan. His research interests were machine learning and deep learning to analyze driving behavior. His current research interests include human-machine interaction and solving the over-trust problem in the driving automation system.

Naoki Akai (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in mechanical engineering from Utsunomiya University, Utsunomiya, Japan, in 2012, 2013, and 2016, respectively. In 2016, he joined the Institute of Innovation for Future Society, Nagoya University, Nagoya, Japan, where he has been a designated Assistant Professor with the Graduate School of Informatics, since 2019. His current research interests include localization, perception, and intelligent systems for mobile robots and autonomous cars. He is a member of the RSJ, SICE, and JSME.

Alexander Carballo (Member, IEEE) received the D.Eng. degree from Intelligent Robot Laboratory, University of Tsukuba, Tsukuba, Japan. From 1996 to 2006, he was a Lecturer with the School of Computer Engineering, Costa Rica Institute of Technology. From 2011 to 2017, he was involved in research and development with Hokuyo Automatic Company Ltd. Since 2017, he has been a Designated Assistant Professor with the Institutes of Innovation for Future Society, Nagoya University, Nagoya, Japan. His main research interests include LiDAR sensors, robotic perception, and autonomous driving.

Eijiro Takeuchi (Member, IEEE) received the bachelor's, master's, and Ph.D. degrees from Intelligent Robot Laboratory, University of Tsukuba, Tsukuba, Japan. From 2008 to 2014, he was with Tohoku University, Japan, as an Assistant Professor. Since 2014, he has been with Nagoya University, Nagoya, Japan, where he is currently an Associate Professor with the Graduate School of Informatics. His main research interests include localization, mapping in robotics, and autonomous driving.

Kazuya Takeda (Senior Member, IEEE) received the B.E. and M.E. degrees in electrical engineering and the D.Eng. degree from Nagoya University, Nagoya, Japan, in 1983, 1985, and 1994, respectively. From 1986 to 1989, he was with the Advanced Telecommunication Research Laboratories, Osaka, Japan. He was a Visiting Scientist with the Massachusetts Institute of Technology, from November 1987 to April 1988. From 1989 to 1995, he was a Researcher and a Research Supervisor with the KDD Research and Development Laboratories, Kamifukuoka, Japan. From 1995 to 2003, he was an Associate Professor with the Faculty of Engineering, Nagoya University. Since 2003, he has been a Professor with the Department of Intelligent Systems, Graduate School of Informatics, Nagoya University and currently is the Head of the Takeda Laboratory, Graduate School of Information Science, Nagoya University. His current research interests include media signal processing and its applications, which include speech recognition, and driving behavior modeling.