Intervention Force-based Imitation Learning for Autonomous Navigation in Dynamic Environments

Tomoya Yokoyama*, Shunya Seiya*, Eijiro Takeuchi*† and Kazuya Takeda*†
* Nagoya University, Nagoya, Aichi, Japan
E-mail: yokoyama.tomoya@g.sp.m.is.nagoya-u.ac.jp Tel/Fax: +8152-789-3647
† Tier IV, Inc, Nagoya, Aichi, Japan
Tel/Fax: +813-5615-8898

***Abstract*—Imitation learning is a data-driven approach that has proven to be successful in building autonomous navigation systems. One of the key tasks in imitation learning is collecting data, but data only collected by humans cannot include many types of data, such as deviating from the target path. If we use a model trained with such data, the deviation will accumulate, and returning to the target path will be difficult. Related studies have demonstrated the importance of correction data, and two types of solutions have been proposed: data augmentation and online sampling. In this paper, we propose a new online sampling method for acquiring correction data that is safe and effective, which uses a device that detects the force applied a steering wheel and accelerator pedal during an intervention. Autonomous navigation experiments are conducted using small vehicles to follow a specified path in static and dynamic environments. Our experimental results show that we can successfully separate intervention data from all collected data using intervention force and that using intervention data for model training is effective for improving the route-following. Also, our model can perform obstacle avoidance and generate appropriate control signals when encountering dynamic objects, such as pedestrians, at specific locations.**

I. Introduction

Countless situations need to be addressed for autonomous vehicles such as self-driving cars, motorized wheelchairs, and small mobile vehicles to operate safely and automatically in dynamic environments. Most fully autonomous navigation systems are designed in a modularized manner [1], [2], and thus are divided into perception, navigation, decision-making, and control components, and contain multiple modules at each stage of the process. For an autonomous vehicle to deal with all of the complex, real-world situations at the operational design domain (ODD), it is necessary to design, test, and integrate each module appropriately, which is expensive.

One approach used to develop autonomous navigation systems is known as imitation learning, where appropriate driving behavior is learned from an expert. The most fundamental method of imitation learning is a type of supervised learning known as behavior cloning, in which, a model learns the relationship between observations and experts' actions from a set of training data [3].

Behavior cloning is effective for some inference problems, but in real-world environments, it may be difficult or impossible for the system to recover when a misalignment occurs [4]. If we consider the case of an expert following a target path, there will generally not be many instances of the expert behavior deviating from the path, and so almost all of the data will be collected along the target path. If a model is trained only using such data, it may not be able to take appropriate action when occurring a deviation from the target path. Therefore, imitation learning systems require not only demonstration data from an expert but also "correction data", which is data showing how to recover from diverging situations. Correction data can be prepared in two methods: data augmentation and online sampling.

Data augmentation is a method of generating data using a framework, such as additional sensors. Bojarski *et al.* [5] proposed a method of imitation learning for end-to-end driving which is an example of behavior cloning using data augmentation. They obtained camera images as input, which were used to generate estimated steering angles as the output. In addition to generating a demonstration dataset, they created correction data for each time step, using data augmentation based on images collected with additional cameras. As a result, their method achieved vehicle control for lane-keeping on real roads.

Online sampling is a method of collecting data while control is being exercised by a trained model. An example of behavior cloning using online sampling is the method developed by Ross *et al.* [3], who proposed an interaction method for imitation learning called the "DAgger", which uses an expert to create training data for a model. In contrast to data augmentation methods, the data collection and model training steps are repeated. During data collection, an expert's correct control over the model's control allows for the acquisition of training data to return from deviations. However, when using the DAgger method, the expert may not perceive their feedback as being sufficient, because the expert's actions are not reflected directly in the control of the vehicle. Therefore, the expert can be exposed to a dangerous situation if the model's output is incorrect during data collection.

In this study, we propose a new online sampling method that uses an intervention device. Intervention forces applied by an expert are used to obtain correction data in the data collection step. Fig. 1 shows an overview of our proposed method. The expert can intervene the actions of the model using the handle of the device. In other words, the handle is controlled automatically during autonomous navigation, but the expert

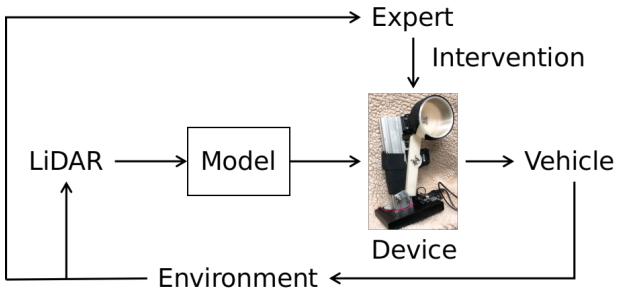


Fig. 1. Overview of our proposed method. We proposed collecting data on force applied by an expert when intervening to correct an autonomous navigation system, and using this data for imitation learning. By using our intervention device, the expert can intervene whenever the model makes a control error.

can interrupt control by applying the force to the handle. In contrast to the DAgger algorithm, the device can override control signals from the model, allowing the vehicle to be controlled instead by the expert's inputs. By installing similar devices in autonomous vehicles, such as self-driving cars, motorized wheelchairs, and small mobile vehicles, experts can interrupt a model when it attempts to carry out dangerous actions, and instantly receive feedback from their intervention actions, allowing them to maintain safe control of the vehicle during data collection.

The contributions of this work are as follows:

1) We propose a new online sampling method that is safe and effective, which uses a device to detect the intervention force.
2) Control force data is collected and forces recorded during expert interventions are used as training data. Objective control signal generation for operating the vehicle is achieved using this small intervention dataset.
3) This method can be used for lane-following, obstacle avoidance, and decision-making in dynamic environments along a target pathway.

The rest of this paper is organized as follows: Section II summarizes related work, Section III describes the proposed method, and Section IV describes our system setup. Section V presents our experimental results, and our results are discussed in Section VI. Finally, in Section VII, we present our conclusions and suggest possible directions for future investigation.

A video of our proposed device and experiments is available at https://drive.google.com/drive/folders/1bM2-uMve-Sm-gA0xTdv5xqii6Lp3E_Ul?usp=sharing.

II. Related work

End-to-end driving is a method that generates a control signal directly from the sensor input. In the following subsections, we discuss several studies related to data augmentation, reinforcement learning, and online sampling that is often used to achieve end-to-end driving, respectively.

A. Behavior cloning with data augmentation

The concept of "end-to-end driving" vehicle navigation was pioneered by Pomerleau et al. [6]–[9]. Their vehicle, called ALVINN (autonomous land vehicle on a neural network), took images from a camera and computed several image transformations by shifting and rotating the original image to the left and right to consider situations where the vehicle had deviated from its intended trajectory. These images were then fed into a back-propagation neural network with 30 outputs to indicate steering direction.

Bojarski *et al.* [5] proposed an autonomous driving method which applied a behavior cloning framework. A Convolutional Neural Network (CNN) [10] model used camera images as inputs and generated a steering angle as the output.

Seiya *et al.* [11] evaluated the effectiveness of Bojarski's data augmentation methods using a small vehicle. They found that the correction data was required for model training and that view-point conversion was the most effective method for robustness. In subsequent studies, Seiya *et al.* [12], [13] used data augmentation to further improve robustness.

Bansal *et al.* [14] proposed another learning-based autonomous driving method which they called ChauffeurNet. Their model learned the relationships between feature maps and future ego positions. This means that the model learned a planning module from human driving data. Their model also used a data augmentation method called Trajectory Perturbation for robustness. Trajectory perturbation is used to create simulated situations such as collisions and deflections from the center of the road which are difficult to obtain from normal driving data. While these data augmentation methods require a particular system to achieve some of their objectives, it will be served multiple objectives simultaneously if the system is not required.

B. Reinforcement learning and inverse reinforcement learning

Inverse reinforcement learning (IRL) is a similar approach to imitation learning. In IRL, the main goal is to learn a reward function from an expert's behavior. A reward function is often used for training agents to learn a policy in reinforcement learning (RL). These methods are used when it is difficult to build the reward function directly, such as in applications like autonomous driving.

Sahand *et al.* [15] proposed the use of an IRL method for autonomous driving. Their model was able to learn a reward function from an expert's demonstrations, allowing an agent to learn human-like lane change behavior from simulations.

Kendall *et al.* [16] and Amini *et al.* [17] proposed practical RL methods for autonomous lane-keeping systems. In their systems, an agent initialized its parameters randomly, and the expert then had to intervene frequently in the control of the agent. Based on this expert demonstration, the system calculated the rewards and trained the agent using RL.

Theerapap *et al.* [18] proposed an RL method that used pull-string angles as a reward function, allowing a robot to learn to follow a specific person in a test environment.

These IRL and RL systems can be understood as an adversarial framework, similar to Generative Adversarial Nets (GANs) [19]. These systems have also been called Generative Adversarial Imitation Learning (GAIL) [20] or Adversarial Inverse Reinforcement Learning (AIRL) [21]. Kuefler *et al.* [22] and Bhattacharyya [23] applied GAIL frameworks for autonomous driving in a simple highway simulation [24]. Li *et al.* [25] introduced visual input into GAIL and demonstrated its effectiveness using a racing game simulation. These IRL and RL methods have challenges in terms of their real-world applications and the need to design reward functions.

C. Online sampling methods

Online sampling methods repeat the sampling and training steps. Laskey *et al.* [26] proposed an online sampling method with a noise injection called DART. Their concept is similar to behavior cloning methods with data augmentation. Noise is added to control signals during data collection to obtain correction data. Codevilla *et al.* [27] also used this method for imitation learning. One drawback of this method is that it can be dangerous in a real driving environment because the control signal includes noise.

Ross *et al.* [3] proposed an online sampling method called DAgger. During sampling, the expert and the model both control the vehicle at the same time, and the control signals from the expert and model are mixed in a probabilistic framework, and the training data is collected. During training, the model learns driving policies from the collected data. When using the DAgger method, the expert does not receive feedback directly, since control signals from the model and the expert are mixed, which can be disorienting for the expert.

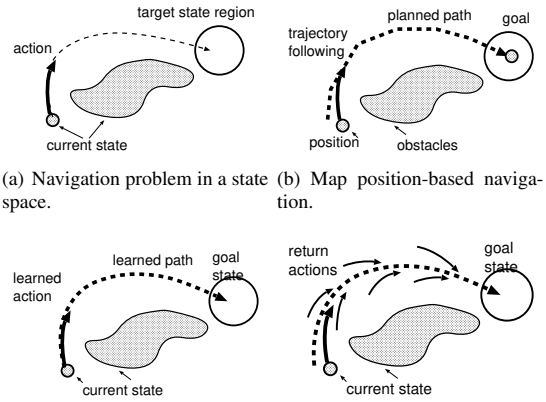
Kelly *et al.* [28] proposed an improved DAgger method called HG-DAgger, by introducing a switch that decides which control signals (the expert's or the model's) are applied at a particular time. In this method, control can be performed without the mixing of expert's and model's control signals, however, it does not take into consideration some device and the possibility that signal changes may suddenly occur at the moment when intervention begins.

III. Intervention force-based imitation learning

In this study, we propose a behavior cloning method that uses a simplified version of HG-DAgger. Our method uses a control force feedback device that can directly obtain correction data from the expert. The device can also obtain control feedback from the vehicle when the expert does not intervene. Our system eliminates the rapid signal changes which may occur at the beginning of an intervention, allowing for safer intervention. In the following subsections, we discuss the details of our proposed method.

A. Preparation of a suitable dataset for imitation learning-based navigation

Behavior cloning methods for autonomous driving require a training dataset for the model to learn the relationships between sensor-inputs and action-outputs. This subsection discusses how to prepare a dataset for imitation learning-based vehicle control.



(a) Navigation problem in a state space.
(b) Map position-based navigation.
(c) End-to-end learning-based navigation.
(d) Correction data to return to the target trajectory.

Fig. 2. Relationship between map position-based navigation and behavior cloning approaches in a state space.

Fig. 2(a) shows a simplified navigation problem in a state space. One basic navigation problem is to find the best action set to reach a destination without a collision. Typical autonomous navigation systems [1], [2] use map position-based navigation (Fig. 2(b)), which estimates positions on a priori map, detects obstacles, plans the shortest path to the destination, and then follows the path using a trajectory controller. In contrast, behavior cloning-based navigation models learn the relationships between current states and the appropriate actions directly from the behavior of an expert human driver (Fig. 2(c)).

Bojarski *et al.* [5] and Seiya *et al.* [12], [13] both realized end-to-end driving using a simple CNN without a recurrent structure or whole path output. This was possible due to Bellman's principle of optimality, which states that a given section of the optimal path from location A to B is consistent with the entire optimal path. In other words, a set of short optimal paths to each position along the optimal path are equivalent to the whole optimal path. Therefore, if a CNN can output the optimal solution for each moment, and the vehicle can be controlled accurately, then the vehicle can follow the optimal path.

Of course, actual vehicle actions can include errors when compared to the optimal solution for several reasons, and as a result, the vehicle can diverge from the optimal path. To solve this problem, Bojarski and Seiya both used a data augmentation technique, in which shifted and rotated position images from a base position, were generated using additional cameras. This facilitates efforts to return to the base position from each shifted position, allowing the vehicle to return to the original trajectory.

Thus, including data to facilitate a return to the target path when the vehicle deviates from its path is important (Fig. 2(d)). In other words, using a dataset that includes correction data is

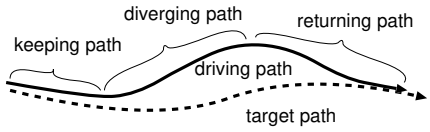


Fig. 3. Types of driving data when an expert intervenes in the control of an autonomous navigation system. The data includes travel along the target route, travel which deviates from the target route, and the path used to return to the target rote after deviating from the route.

highly desirable for imitation learning-based navigation.

B. Correction data using intervention force

Researchers have achieved trajectory following through end-to-end driving, however various navigation problems remain, such as dynamic obstacle avoidance and velocity control. Some of the problems faced by conventional data augmentation approaches are that the special system needs to be designed and that the dataset tends to be very large since data augmentation is performed for every scene.

A simpler approach is to extract correction data from driving data. When an autonomous vehicle behaves in an unintended manner, humans can intervene and provide the input necessary for correction. As a result, the driving data includes the recovery actions needed for various problem situations (which are good data for learning) as well as the faulty actions which resulted in divergence from the target path (which are not good for learning). Fig. 3 illustrates these types of driving data. The dashed line represents the target path, while the solid line represents the vehicle's actual path when trying to follow the target path. The driving data thus includes three states: keeping the target path, diverging from the target path, and returning to the target path. If a learning method uses all of this driving data, including divergence data, the learning results will sometimes guide the vehicle off the target path, therefore the extraction of data from path keeping activity and activity when returning to the path is important for imitation learning.

Correction data is obtained when a human intervenes in automatic control and this intervention can be detected from the force applied to the steering wheel and accelerator pedal. This idea is based on power steering in cars, which uses a motor to magnify the force applied to the steering wheel. When using power steering, the force from the motor can be detected, so the correction data is obtained from the forces applied during the intervention, while the data which is not needed for learning can be removed.

C. Proposed algorithm

In this subsection, we describe our proposed online sampling method which uses the forces applied by an intervention device, which is also described in Algorithm 1. Firstly, an expert directs a vehicle using the input device to collect the first dataset D_0, which includes input-sensor data and output-action data. Subsequently, during learning, the first model π_0 is trained from scratch using dataset D_0. Next, the expert driver begins the data collection step, which is tracked chronologically using timestep t, during which trained model π_s primarily controls the vehicle, with the control state is reflected in the input device. If the expert applies an intervention force f_t at timestep t to the input device, the expert controls the vehicle. During the intervention, the proposed system collects the expert intervention data D_{expert}. Finally, after the data collection step, the training step is initiated, during which the dataset D_{expert} is added to D_s to create a new dataset D_{s+1}, and the new model π_{s+1} is trained using D_{s+1} from scratch. The collection and training steps are repeated N times.

Algorithm 1 Our proposed algorithm

```
Collect initial dataset D_0
Train initial model π_0 using D_0
for repeating step s = 0, 1, ..., N do
  for driving timestep t = 0, 1, ..., T do
    Control vehicle using model π_s
    if expert applies force f_t > ε to device then
      Control vehicle by expert
      Collect expert dataset D_expert
    end if
  end for
  D_{s+1} ← D_s ∪ D_expert
  Train model π_{s+1} using D_{s+1}
end for
```

Our proposed method offers three advantages Compared to the methods proposed in related studies. Firstly, our method does not use a physical model to obtain correction data. Therefore, the method can be extended to various navigation situations such as obstacle avoidance and dynamic environments. Secondly, the total amount of intervention data used for training the model is small, therefore the total size of data for learning to support various situations is reduced. Finally, the intervention device allows for smooth transitions between autonomous and intervention modes, therefore the driver can maintain safe and comfortable travel through proactive intervention.

IV. System structure

A. Intervention device

The device used in this study for expert intervention is shown on the right of Fig. 4. Our device is composed of a wheel for steering control and a lever for velocity control. When the vehicle is being controlled automatically, control components of the device receive feedback from the vehicle and are controlled automatically. Since devices such as gamepads do not provide feedback, sudden changes in signals can occur at the beginning of an intervention, but this does not occur when using our device. The expert can intervene in the autonomous system and control the vehicle directly when applying force to the device. The force signals are saved during the intervention and they are used for extracting the intervention data.

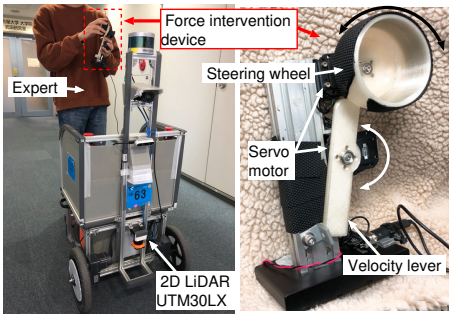


Fig. 4. Experimental setup for basic experiments. The figure on the left shows our experimental vehicle and the figure on the right is a close-up of our intervention device, which consists of a steering wheel and a velocity lever. The device receives feedback from the autonomous vehicle and is controlled automatically. The expert can intervene in the control by applying force to the device.

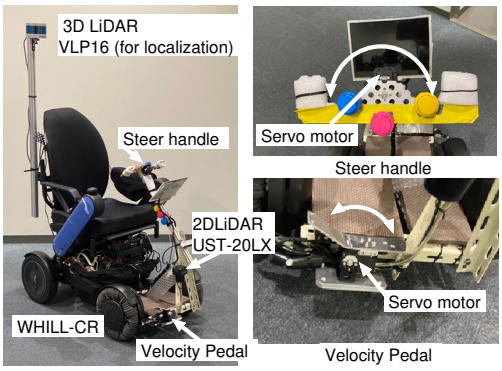


Fig. 5. Experimental setup for our advanced experiment. The expert can intervene the in control of the autonomous driving system using the steering handles and velocity pedal in a manner similar to that used by the autonomous vehicle.

For our experimental evaluation, we constructed vehicles equipped with several sensors. For our basic experiments, which consisted of following a specified route, static obstacle avoidance, and decision-making in dynamic environments, we used the vehicle shown on the left of Fig. 4. For our advanced experiment, which involved outdoor navigation, we used the device shown in Fig. 5, which was designed to be easily overridden by an expert, allowing for interventions to be performed with little force.

To obtain information about the surrounding environment, we used a single-line (2D) LiDAR, for obtaining point clouds to create grid maps. The vehicle was controlled by either control signals from the model or an expert, who monitored the movement of the vehicle and who could intervene in the control of the model at any time by using the device.

Fig. 6 shows an example of an intervention force. The blue and orange lines show the force of the intervention and the angular velocity of steering, respectively. Angular velocity is controlled by the model when the vehicle is under automatic control, but when intervention occurs the control signal is switched seamlessly and smoothly to an expert. By noting the

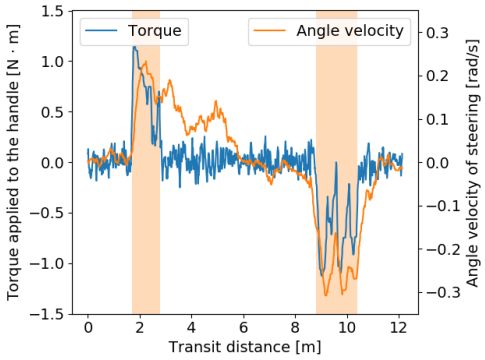


Fig. 6. Example of an intervention force. When the expert intervenes, the applied force increases sharply (blue line) and angle velocity changes (orange line). Therefore, intervention data can be separated from other data (light orange areas).

period when the force applied to the device exceeds a certain value, intervention data can be easily distinguished from other data, as shown in the light orange areas.

B. Sensor data arrangement for network input

We used a grid map representation for observation o_t of the surrounding environment. Fig. 7(a) shows an example of a grid map with $H \times W$ cells (a height of H cells and a width of W cells). The actual H and W values are shown in Table. I. The white cells represent occupied areas where the LiDAR data indicated an environmental obstacle, while the black cells represent open space.

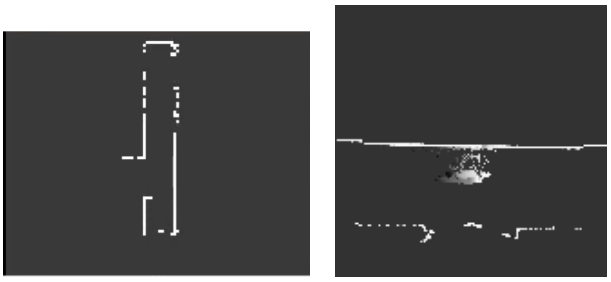
To support operation in dynamic environments, we used a blurred motion-grid map (Fig. 7(b)), which contains an accumulation of a few seconds of LiDAR data. In the figure, grayscale values of the cells indicate the age of scans, with the color of the older dynamic cells turning black as time passes. To generate the grid map, each set of scanner data is mapped to world coordinates, accumulated, and then re-converted into sensor coordinates. Thus, the values in the grid map change with the passage of time and the currently unoccupied cells turn black with the passage of time. In this manner, static obstacles are mapped in white and moving obstacles have grey tails (representing old data).

C. Network structure

Our neural network model is comprised of five convolutional layers and five fully-connected layers to learn these relationships, following a previous study [11]. The input of the model is the grid map and the outputs are the velocity v and angular velocity ω of steering. The convolution kernel size is five in the top three layers, and three in the last two layers.

V. Experiments and Results

To evaluate our proposed method, we conducted four experiments; following a specified route, avoidance of a static obstacle, decision-making in a dynamic environment, and



(a) Grid map (b) Blurred motion grid map

Fig. 7. Example of a grid map (left) and a blurred motion grid-map (right). The white cells have a value of 1, while the black cells have a value of 0. The grid map represents the surrounding environments, while the blurred motion grid map distinguishes between static objects and dynamic objects.

Fig. 8. Experimental environment for specified route-following. We attempted to follow a specified, indoor route, as shown by the red line. The route included two turns and a narrow gate.

outdoor navigation. Table. I shows the conditions for these experiments.

A. Route-following

This experiment investigates whether robust learning-based autonomous navigation can be achieved by collecting the data from the scenes that involve expert intervention. In this experiment, we trained two models with different amounts and types of data. One model was trained using all of the collected data, including intervention data, while the other model was trained using intervention data only. We compared both route following performance and the amount of data utilized for these two models. To evaluate performance, we measured the amount of intervention force that needed to be applied by the expert to the steering handle to stay on course. The less force the expert needed to apply, the better the model's performance.

Fig. 8 shows the environment for our route-following experiment. The red line shows the specified route to be followed by the vehicle, which includes two corners and a narrow gate. The width of the vehicle is 60 cm while the width of the narrow gate is only 85 cm. In terms of control, the following task is challenging.

Fig. 9 shows the results of this experiment. The orange and blue lines show the amount of training data used and the average amount of torque of every frame applied for the steering. The average torque of every frame represents the intervention force needed to reach the destination without a collision or going in the wrong direction. In other words,

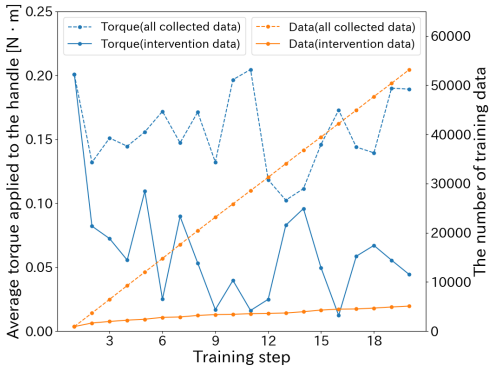


Fig. 9. Average applied torque in relation to the amount of training data. Increasing the number of sampling steps decreased the average force required. When all of the data was used for training, the average force did not decrease.

the smaller the force needed, the more accurately the vehicle was able to operate on its own. In terms of the amount of training data used, we collected over 50,000 frames of driving data in total during 20 sampling steps for training the "all data" model. We also succeeded in separating just under 10,000 frames of intervention data from this data for training the "intervention data only" model. In terms of navigation performance, the amount of intervention force needed to stay on course decreased overall when using the model trained with only intervention data, as the number of repeated data collection and training steps increased. However, the amount of intervention force needed did not decrease when we used all of the collected data.

Fig. 10(a) shows where interventions were needed to correct the output of the model while data collecting step. The green points correspond to the target path and the purple lines denote the locations where interventions occurred. Intervention tended to be needed when the direction of travel changed abruptly. The trajectories when the intervention was not needed for reaching the destination are shown in Fig. 10(b).

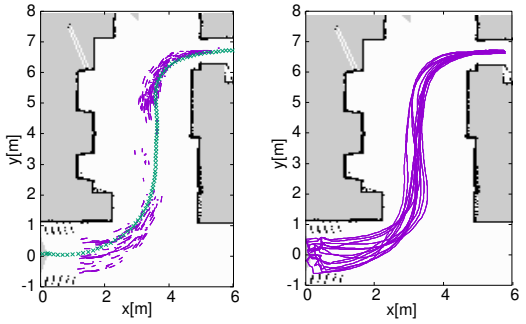
After 20 trails of repeating step, we also compared the number of times the vehicle could reach its destination without collision with a wall, in the absence of any expert intervention. When all of the collected data was used for training a model, the vehicle was able to reach its destination 0 times out of 15 attempts. In contrast, when only intervention data was used for training a model, the vehicle was able to reach the destination in 13 out of 15 attempts.

B. Static obstacle avoidance

We expanded the usage of our method to include obstacles. In this experiment, an obstacle was randomly placed on a straight section of the designated path, using the same environment as in the previous experiment. The size of the obstacle was 53 cm in length, by 31 cm in width, by 43 cm in height. The expert collected data to avoid the obstacle, as it was placed in various locations in each trail. After training the model using 10,226 frames of data from about 40 obstacle avoidance trails, we evaluated whether or not the vehicle could

TABLE I
EXPERIMENTAL CONDITIONS

	Specified route-following	Static obstacle avoidance	Decision-making in dynamic environments	Outdoor navigation
Number of frames of training data in final step	5,097(intervention data) 53,165(all data)	10,226	3,747	21,235
Input map	Grid map	Blurred motion grid map		
Number of cells in input map	$80 \times 100 \times 1$	$150 \times 150 \times 1$		
Output signal	Steering	Steering and velocity		



(a) The green points represent the target trajectory, and the purple lines show where interventions occurred.

(b) Trajectories of the vehicle during autonomous navigation. These trajectories were generated using the model trained using only intervention data.

Fig. 10. Trajectories of the vehicle when following the specified route.

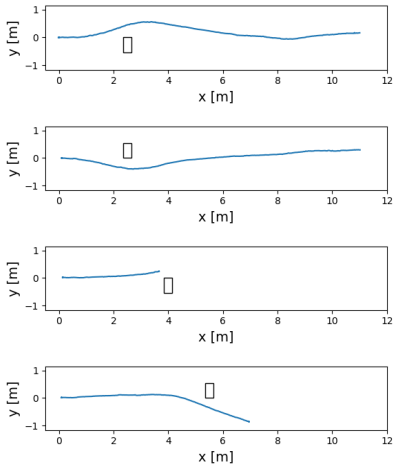


Fig. 11. Examples of trajectories during static obstacle avoidance experiment. Lines represent the trajectory of the vehicle and the black rectangles are obstacles. The top two trajectories are examples of successful obstacle avoidance, while the bottom two are examples of unsuccessful attempts.

automatically avoid a collision with a wall or with an obstacle placed in ten different locations.

Our results show that the vehicle was able to automatically follow the designated route without a collision in 6 out of 10 trails. Fig. 11 shows some of the trajectories. The first two figures show how the vehicle could avoid the obstacle when approaching it, and could then return to the center of the path after passing the obstacle. In contrast, the last two figures show trails where the vehicle collided with the obstacle or a wall.

C. Decision-making in dynamic environments

We also considered a situation that is likely to occur in dynamic environments to investigate the difference in the velocity output by the models when encountering a stationary pedestrian versus a pedestrian moving towards the roadway. To simplify the experiment, the accelerator lever of the intervention device was activated while the vehicle is in a stationary state, so that the vehicle was not actually moving. During data collection, the expert decelerated from 0.3 $\mathrm{m/s}$ when a pedestrian approached the path of the vehicle and accelerated after the pedestrian had passed. On the other hand, when a pedestrian was standing near the intersection but remained stationary, and did not move to cross the street, the expert did not slow down. The experiment was conducted in the same manner as the previous route-following experiment, and 3,747 frames of intervention data were used for training. We evaluated the output velocity for the vehicle when the pedestrian was moving and when the pedestrian was not moving.

Fig. 12 shows the results of this experiment. The blue and the orange lines show the velocity output by the model when the pedestrian was moving and when not moving, respectively. The vehicle slowed down when the pedestrian was approaching and crossing the vehicle's path and increased after the pedestrian had crossed. The velocity of the vehicle remained high and did not change greatly when the pedestrian remained stationary near the intersection. Fig. 13 shows input images when moving and stationary pedestrians are located at the same point near the intersection. At this particular location, the velocity of the vehicle when a pedestrian was moving (image on the left) was about 0.03 $\mathrm{m/s}$, and when the pedestrian was not moving(image on the right) the vehicle's velocity was about 0.25 $\mathrm{m/s}$.

D. Outdoor navigation

As an integration of the previous three experiments, we then attempted to perform outdoor navigation with our model, using the vehicle shown in Fig. 5. Fig. 14(a) shows a bird's -eye view of the environment used in this experiment, which included some pedestrians and intersections. The expert drove 14 laps around the course and 21,235 frames of training data were collected.

The vehicle could navigate the course successfully by itself

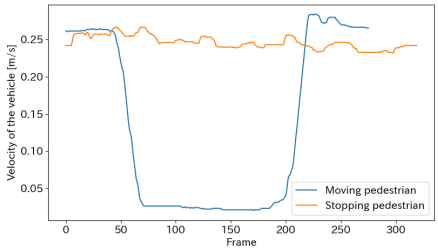


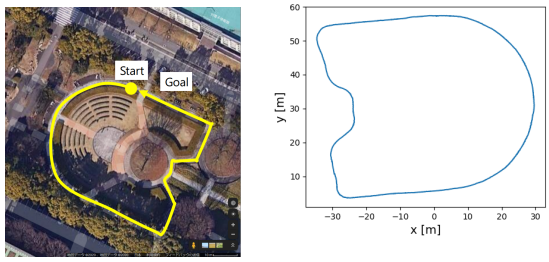
Fig. 12. Change in velocity when moving and stationary pedestrians were present. The blue and the orange lines show velocity output when a pedestrian in front of an intersecting road is moving or stopped. When the pedestrian was moving, the vehicle reduced its speed. In contrast, when the pedestrian was not moving, the vehicle maintained its velocity.

Fig. 13. Comparison of moving and stationary pedestrian motion at the same location. The representation of the moving pedestrian (left) shows a gray tail to the left, which represents movement from the pedestrian's prior location. The representation of the stationary pedestrian (right) has no tail, including no movement from a previous location.

(without intervention) when the model was trained with the 14 laps of intervention data. The trajectory is shown in Fig. 14(b). Fig. 15 shows how the differences in the vehicle's trajectory depending on whether or not a stationary pedestrian was present on the road at the same location. The pedestrian was standing at a point with an x value of approximately -15 m. The blue line shows the vehicle's trajectory when there was no pedestrian (the vehicle drove straight), while the orange lines show the vehicle's trajectories when a pedestrian was present (the vehicle swerved to avoid the pedestrian). Fig. 16 shows how the vehicle velocity differed depending on the presence of a pedestrian crossing at an intersection. The blue and orange lines denote vehicle velocity when there was no pedestrian and when there was a pedestrian, respectively. Even when there was a pedestrian crossing the road, sometimes the vehicle did not slow down.

VI. Discussion

In the route-following experiments, our results show that the intervention force needed for the vehicle to remain on course decreased when only using the intervention data for model training. These results indicate that the use of intervention data is effective for improving the robustness of autonomous driving. By extracting and using the intervention data for training, data from deviant actions was excluded, and thus



(a) Bird's-eye view of the vehicle's route.

(b) Trajectory of the vehicle after training from 21,235 frames of intervention data.

Fig. 14. Overview of outdoor navigation experiment

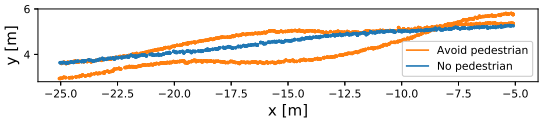


Fig. 15. Comparison of trajectories when there was a pedestrian in the path of the vehicle (orange lines), and when there was no pedestrian (blue line). When a pedestrian was present, the vehicle swerved to avoid a collision. When there was no pedestrian, the vehicle continued along the route.

the vehicle was better able to travel without deviations. In addition, Fig. 10(b) indicates that the trajectories of the vehicle tended to expand when there was a wide path and to converge when the path was narrow. This suggests that the subtlety of control by the model may vary depending on environmental differences such as road width.

In the experiments involving indoor obstacle avoidance, our results show that our method can be applied not only to simple route-following but also in scenarios that include changing environmental conditions. This indicates that the

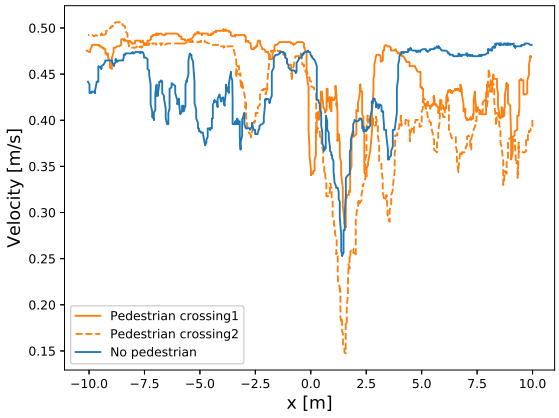


Fig. 16. Change in velocity depending on the presence of pedestrian at the intersection. The blue and orange lines show vehicle velocity when there was no pedestrian present and when there was a pedestrian crossing the intersection, respectively.

(a) (b) (c)

Fig. 17. Input grid maps of the intersection where the data shown in Fig.16, showing the moment when the vehicle's velocity was at its minimum for each of the three experimental conditions: (a) blue line = no pedestrian, (b) solid line = pedestrian crossing quickly, and (c) orange dashed line = pedestrian crossing slowly.

CNN was able to abstract features, allowing the vehicle to avoid the obstacles. As a result, the vehicle was able to avoid the obstacles automatically, even when they appeared outside the collected data. Sometimes the vehicle failed to avoid an obstacle even though it reacted to the obstacle because the change in the steering angle was not sufficient to avoid it. Similarly, the vehicle sometimes failed to return to the path when it missed the obstacle. To address these problems, we consider that adjusting the range of input data being used, depending on the objective, seems to be effective. Our input grid map represented an area of $16\mathrm{m} \times 20\mathrm{m}$, but this may be too large an area, because some obstacles become relatively small. This suggests that extracting important features in order to avoid obstacles is difficult if the size of input data is not adequate.

In the experiments conducted in a dynamic environment, our results show that our method could distinguish between moving and non-moving obstacles. Therefore, our proposed method can be used in complex situations such as dynamic environments.

In the outdoor navigation experiment, we observed that when pedestrians were crossing at an intersection, the output speed of the autonomous vehicle differed depending on the speed of the pedestrian. Fig. 17 shows an input grid map of the intersection. By comparing Fig. 17(b) and Fig. 17(c), we can see that the speed the pedestrians are traveling is different; the pedestrian in the former is moving faster than the pedestrian in the latter. When examined in conjunction with Fig. 16, it appears that the vehicle does not slow down significantly when the pedestrian is moving quickly, possibly because the pedestrian is expected to finish crossing the road before the vehicle arrives. In contrast, the vehicle does slow down when the pedestrian is moving slowly, because maintaining the higher velocity could result in a collision.

VII. Conclusions

In this paper, we have proposed new online data sampling methods for autonomous driving using an intervention device. Since the device proposed in this study can receive feedback on the control of the autonomous vehicle, the expert can intervene smoothly if needed. Furthermore, our method provides an easy and safe way to collect training data from an expert. Our results show that using the collected expert intervention data for training is an effective method of improving the robustness of an agent's driving ability. Our experiment demonstrated that our method can be applied to route-following, obstacle avoidance along a specified route. It is unclear, however, if our method will work when following a different route than the one used for training.

In future work, we will need to evaluate our method in real-world driving scenarios using real automobiles. In order to improve performance, we are looking to make use of data from scenes where intervention has not occurred and to be able to notice obstacles. In addition, although this study dealt with navigation on a specific pathway, it will be extended to navigation on any pathway. Furthermore, we will also evaluate whether or not model-based systems, which are designed in a modularized manner, can be improved, in terms of safety and driving comfort, by applying our method.

References

[1] Shinpei Kato, Eijiro Takeuchi, Yoshio Ishiguro, Yoshiki Ninomiya, Kazuya Takeda, and Tsuyoshi Hamada, "An Open Approach to Autonomous Vehicles," in *Journal of IEEE Micro* vol. 35, no. 6, pp. 60-68, 2015.

[2] Baidu Apollo, https://github.com/ApolloAuto/apollo, [Accessed: 2020/06/15]

[3] Stephane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell, "A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning," in *Proc. of International Conference on Artificial Intelligence and Statistics* (*AISTATS*), pp. 627-635, 2011.

[4] Stephane Ross and J. Andrew Bagnell, "Efficient Reductions for Imitation Learning," in *Proc. of International Conference on Artificial Intelligence and Statistics* (*AISTATS*), pp. 661-668, 2010.

[5] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba, "End to End Learning for Self-Driving Cars," *arXiv preprint arXiv:1604.07316*, 2016.

[6] Dean A. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network," in *Proc. of Advances in Neural Information Processing Systems 1 (NIPS)* pp. 305-313, 1998.

[7] Dean A. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," in *Neural Computation* vol. 3, no. 1, pp. 88-97, 1991.

[8] Dean A. Pomerleau, "Knowledge-based training of artificial neural networks for autonomous robot driving," in *Robot Learning* pp. 19-43, 1993.

[9] Todd M. Jochem, Dean A. Pomerleau, and Charles E. Thorpe, "Vision-based neural network road and intersection detection and traversal," in *Proc of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 'Human Robot Interaction and Cooperative Robots'* vol. 3, pp. 344-349, 1995.

[10] Yann Lecun, Bernhard Boser, John Denker, Don Henderson, R.E. Howard, W.E. Hubbard, and Larry Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," in *Neural Computation* vol. 1, no. 4, pp. 541-551, 1989.

[11] Shunya Seiya, Daiki Hayashi, Eijiro Takeuchi, Chiyomi Miyajima, and Kazuya Takeda, "Evaluation of Deep Learning-Based Driving Signal Generation Methods for Vehicle Control," in *Proc. of Fast-ZERO International Symposium* (*Fast-ZERO*), 2017.

[12] Shunya Seiya, Alexander Carballo, Eijiro Takeuchi, Chiyomi Miyajima, and Kazuya Takeda, "End-to-End Navigation with Branch Turning Support using Convolutional Neural Network," in *Proc. of IEEE International Conference on Robotics and Biomimetics* (*ROBIO*), 2018.

[13] Shunya Seiya, Alexander Carballo, Eijiro Takeuchi, and Kazuya Takeda, "End-to-End Driving using Point Cloud Features," in *Proc. of Fast-ZERO International Symposium* (*Fast-ZERO*), 2019.

[14] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale, "ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst," *arXiv preprint arXiv:1812.03079*, 2019.

[15] Sahand Sharifzadeh, John Chiotellis, Rudolph Triebel, and Daniel Cremers, "Learning to Drive using Inverse Reinforcement Learning and Deep Q-Networks," in *Proc. of Advances in Neural Information Processing Systems* (*NIPS workshop*), 2016.

[16] Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda , John-Mark Allen, Vinh-Dieu Lam, Alex Bewley and Amar Shah, "Learning to Drive in a Day," in *Proc. of IEEE International Conference on Robotics and Automation* (*ICRA*), 2019.

[17] Alexander Amini, Igor Gilitschenski, Jacob Phillips, Julia Moseyko, Rohan Banerjee, Sertac Karaman, Daniela Rus, "Learning Robust Control Policies for End-to-End Autonomous Driving From Data-Driven Simulation," in *IEEE Robotics and Automation Letters* vol. 5, no. 2, pp. 1143-1150, 2020.

[18] Pasin Theerapap, Yasuo Hayashibarar, and Yuichi Ueda, "Study on an Autonomous Mobile Robot using a Reinforcement Learning: - Study on acquisition of behavior following a person, while keeping a certain distance -," in *Proc. of JSME annual Conference on Robotics and Mechatronics* (*Robomec*), 2018.

[19] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative Adversarial Nets," in *Advances in Neural Information Processing Systems* (*NIPS*), 2014.

[20] Jonathan Ho and Stefano Ermon, "Generative Adversarial Imitation Learning," in *Advances in Neural Information Processing Systems* (*NIPS*), 2016.

[21] Justin Fu, Katie Luo, Sergey Levine, "Learning Robust Rewards with Adversarial Inverse Reinforcement Learning," in *Proc. of International Conference on Learning Representations* (*ICLR*), 2018.

[22] Alex Kuefler, Jeremy Morton, Tim Wheeler, and Mykel Kochenderfer, "Imitating Driver Behavior with Generative Adversarial Networks", in *Proc. of International Conference on Intelligent Vehicle* (*IV*), 2017.

[23] Raunak P. Bhattacharyya, Derek J. Phillips, Blake Wulfe, Jeremy Morton, Alex Kuefler, and Mykel J. Kochenderfer, "Multi-Agent Imitation Learning for Driving Simulation," in *Proc. of International Conference on Intelligent Robots and Systems* (*IROS*), 2018.

[24] James Colyar and John Halkias, "US highway 101 dataset," *Federal Highway Administration (FHWA), Tech. Rep.FHWA-HRT-07-030*, 2007.

[25] Yunzhu Li, Jiaming Song, and Stefano Ermon, "InfoGAIL: Interpretable Imitation Learning from Visual Demonstrations," in *Neural Information Processing Systems* (*NIPS workshop*), 2017.

[26] Michael Laskey, Jonathan Lee, Roy Fox, Anca Dragan, and Ken Goldberg, "Dart: Optimizing noise injection in imitation learning," in *Proc. of the 1st Annual Conference on Robot Learning* (*CoRL*), 2017.

[27] Felipe Codevilla, Matthias Muller, Antonio Lopez, Vladlen Koltun, and Alexey Dosovitskiy, "End-to-end Driving via Conditional Imitation Learning," in *Proc. of IEEE International Conference on Robotics and Automation* (*ICRA*), 2018.

[28] Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J. Kochenderfer, "HG-DAgger: Interactive Imitation Learning with Human Experts," in *Proc. of International Conference on Robotics and Automation* (*ICRA*), 2019.