2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI)

2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI) | 978-1-6654-1236-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/SWC50871.2021.00029

A Lane Changing Model Based on Imitation Learning and Gaussian Velocity Fields

Ming Wei
School of Computer Science
China University of Geosciences
Wuhan, P.R. China
mingwei@cug.edu.cn

Wei Ren*
School of Computer Science, China University of Geosciences
Wuhan, P.R. China
Henan Key Laboratory of Network Cryptography Technology
Zhengzhou, P.R. China
Key Laboratory of Network Assessment Technology, CAS
(Institute of Information Engineering, Chinese Academy of Sciences)
Beijing, P.R. China
weirencs@cug.edu.cn

***Abstract*—Making lane-changing decisions for autonomous vehicles has been widely regarded as a core task for autonomous driving. In this paper, we proposed an imitation learning framework to navigate a target vehicle through congested highway scenarios. In this framework, to efficiently capture the dynamic traffic environments, a representative feature named Gaussian Velocity field which is invariant to the number of surrounding obstacles is introduced. Then we train a GVF-CNN model to determine the actions of the target vehicle among a set of five categories which helps autonomous vehicles make decisions. Simulations are conducted in top-down bird-eye view and demonstration data are collected from two ways: (a) highD dataset, a real-world collected highway dataset for pre-training the model, and (b) simulation data for fine-tuning the model. By adopting the well-trained model to control the target vehicle in the simulator to conduct the simulation, the experiment results demonstrate the effectiveness of our method. Finally, detailed analysis of crash reasons as well as the potential improvement methods are discussed.**

***Index Terms*—Autonomous vehicles, Imitation learning, Lane-change, Gaussian Process, GVF-CNN model**

I. Introduction

With the popularity of vehicles and traffic jam appearing on highways, the safety of driving is of paramount importance. Drivers may not be able to deal with the complex situation perfectly in time due to many reasons. Thus, autonomous driving is becoming an important research field nowadays, and route planning becomes a core issue in the field of autonomous driving.

There exist many researchers dealing with the route planning issue. Kasper et al. [1] used Bayesian networks to make decisions for highway vehicles. However, its applicability is limited because the number of lanes of the highway should be defined before and only the nearest well-observed vehicles would be considered. Zhang et al. [2] proposed a specified structure and trained a neural network to help make decisions for vehicles. But they only considered the nearest obstacle at each timestamp. Leonhardt et al. [3] trained a model using features of close obstacles both on the current lane and nearby lanes. However, these are different from real situations in daily life. Because human drivers always consider nearby environment e.g., obstacles as an entirety and make an overall decision. Moreover, it is more intelligent not to separate space factors from time factors. For example, Do et al. [4] only considered space factors in which they designed a grid-based model. On the contrary, Galceran et al. [5] propsed a method using temporal features while ignoring spacial features. In common sense, drivers prefer to make decisions considering the environment within a region of interest (ROI). In our experiments, with the target vehicle as the center point, an area of 40 meters ahead or following and 6 meters in the left and right sides is defined as the ROI. Therefore, we propose a framework (as shown in Fig. 1) that can make decisions for autonomous vehicles which uses the concept of Gaussian Velocity field (GVF) to view the nearby environment as a whole part.

The aim of this paper is to design a model that can make the autonomous vehicles navigate through congested highways. In this paper, to make adaptive decisions in dynamic traffic environments, we manipulated a learning-from-demonstration method (as shown in Fig.1) based on *highway-env* simulator [6]. In Fig.1, the blue lines represent the training part. First, we adopt highD dataset to generate some real-world demonstration as main training data, and develop a traditional Convolutional Neural Network (CNN). The red lines in Fig.1 represent the simulation part. To make the CNN adapt to the simulator, we then adopt a built-in planning methods, *i.e.*, optimistic planning of deterministic systems (OPD) [7], to generate a small amount of training

*W. Ren is the corresponding author.

978-1-6654-1236-0/21/$31.00 ©2021 IEEE
DOI 10.1109/SWC50871.2021.00029

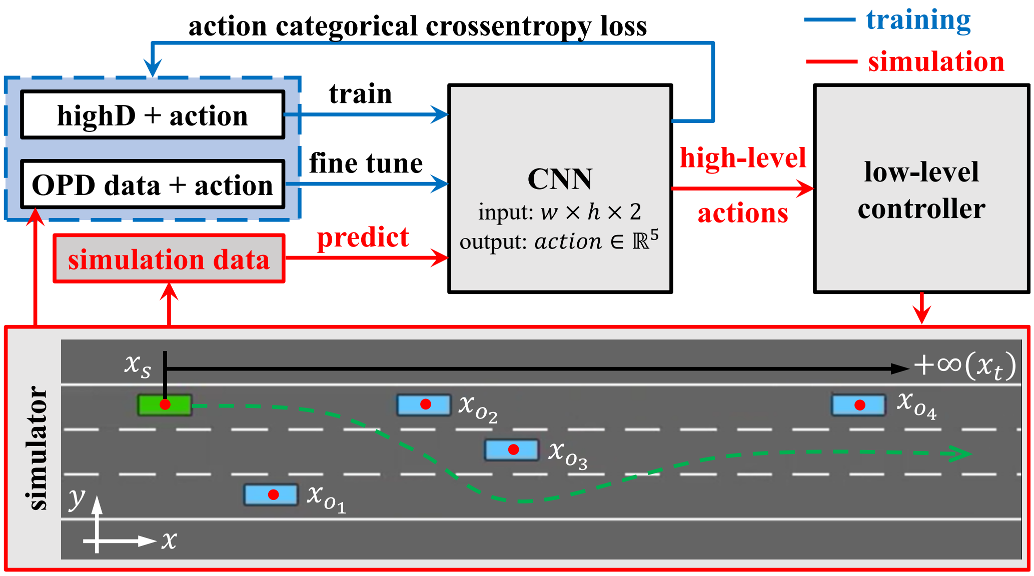


Fig. 1: The framework of learning from demonstration. In the CNN module, the parameters w and h refer to the width and height of the Gaussian Velocity field, and c refers to the number of channels.

data for fine-tuning the model. In this way, we obtain a well-trained model which maps the observations of the traffic environments to action sets. The contribution of this paper is as follows:

1) A deep learning framework via learning from demonstrations method is proposed, which helps vehicles make decisions while driving through congested highways.
2) A 3D-GVF is developed to improve the original 2D-GVF, which can avoid the problem of missing observations when the other vehicles are with near-zero relative speed.
3) A GVF-CNN model is established, which adopts the extracted GVF to make planning decisions, the model is demonstrated to be with good performances.

The rest of the paper will be organized as follows: Section II introduces the related works. Section III introduces the features and network architecture of our imitation learning methods. Section IV demonstrates the training procedures and simulation experiments, results and failure cases analysis are discussed as well. Section V concludes the paper.

II. Related Works

For many years, researchers have been studying the mechanisms of lane changing for the practicality and promotion of autonomous driving. Dogan et al. [8] used a recurrent neural network and a set of support vector machine (SVM) to predict lane changes in 2011. But its real-time performance is not very satisfactory and the choosing of appropriate features has a decisive influence on the experimental results. In addition, the security of parameters during the training process is also of great significance [9]. Vallon et al. [10] introduced a modelling method to classify drivers' decisions, they used a SVM based classifier to train personalized models which helps improve driving experience. This method made progress in showing preferences of different drivers. Nazari et al. [11] used an end-to-end framework which used reinforcement learning to solve route planning problems and observed the reward signals. This method achieved great performance with competitive solution-time. Wang et al. [12] also used deep reinforcement learning to solve lane changing problems. Li et al. [13] used hidden Markov decision process to help make decisions in the decision making process.

In order to retrieve useful information from environments and make proper decisions for changing lanes while interacting with other vehicles efficiently, a variety of representatives were selected in state-of-the-art such as relative distance/speed/acceleration between the ego vehicle and the target vehicle. These features can be directly measured using sensors. For example, Xie et al [15] used deep learning methods to help solve lane-changing problems and found that the most important factor was the relative position of preceding vehicles. They used deep belief network and long short-term memory (LSTM) to train models and got high accuracies. Alao, Zhang et al. [28] used the same method LSTM neural networks to extract features of six surrounding vehicles to make lane-changing decision. However, the dimension of these features is sensitive to

TABLE I: Existing methods about lane changing problems and their feature

Literature	Traffic representations	varying-dimension problem
[4], [14], [15]	velocity, relative velocity, relative distances	yes
[16]	GPS/IMU	yes
[8]	lane offset, derivative of lane offset, lateral acceleration, steering angle, TTC, curvature	yes
[10]	relative distances, relative longitudinal velocities	yes
[17]	distance from the centerline, lateral velocity, the potential feature	yes
[18]	position, demand	yes
[19]	relative distances, longitudinal acceleration	yes
[20]	relative trajectories	yes
[1], [12]	occupancy grid cell, relative velocity	no
[21]	occupancy grid cell	no
[22]	drivability cells	no
[23], [24]	safety potential field	no
[25], [26]	Gaussian velocity field	no
[27]	semi-stochastic potential fields	no

the number of traffic agents in the environment, which we refer to as the time-varying dimension problem. We can see that these researches are done in restricted experiment environment and need previous knowledge of the number of lanes and so on. Mahajan et al. [18] combined SVM and LSTM to predict lane-changing maneuver classes and made comparisons with other deep learning methods. Furthermore, Liu et al. [29] applied drivers' previous memory and the information of surrounding vehicles' previous paths to the training and accessing of the model. All in all, there are few researches about lane-changing algorithms without knowing the number of lanes and other background knowledge.

We explicitly list some literature as well as their selected traffic representations in Table 1. From the table, we can conclude that the time-varying dimension problem widely exists in many literature. However, we also observe that one kind of field-like feature would tackle this problem. For instance, the occupancy grid cell [21], the drivability cells [22], the safety potential field [23], [24], the Gaussian velocity field [25], [26], and the semi-stochastic potential fields [27].

III. METHODS

As has been discussed above, existing road planning methods always need many parameters about environment around vehicles and the specific environment is very strict. In order to solve this problem, in the literature [25], [26], a GVF is constructed based on Gaussian processes, which can depict the dynamic motion trends of a set of surrounding vehicles with a fixed size of tensor, while the number of vehicles in this set may be time-varying. In our paper, we refer to this two-dimension Gaussian Velocity field as 2D-GVF. In this section, we will follow the notations of the 2D-GVF and briefly revisit the construction of 2D-GVF. Then, we will develop an improved GVF, called 3D-GVF to extract more sufficient traffic representations. Furthermore, based on these GVF representations, we will develop a deep learning model called GVF-CNN. To this end, we then construct the imitation learning model as a general framework which is shown in Fig. 1.

A. Traffic Representation Learning

Generally, algorithms tend to be with good performances if fed with data that captures efficient features. Here we focus on a traffic representation learning method to extract efficient traffic features that can depict the driving environment. To tackle the aforementioned time-varying dimension problem, we introduce and improve the GVF, which regards the position and speed information of obstacles around the target vehicle as a whole field with velocity.

1) Conventional Gaussian velocity field (2D-GVF): This model is a Gaussian process regression model, with the relative locations and velocities as input and a couple of mean functions as output. Specifically, given the locations of N vehicles $\{x_i, y_i\}_{i=1}^N$ and the velocities $\{v_{x_i}, v_{y_i}\}_{i=1}^N$ of N vehicles within the observation region, one can construct the regression model based on Gaussian process to obtain the mean function, i.e., velocity field. Here we first focus on the x direction and define the kernel of Gaussian process using a standard squared exponential covariance functions:

$$K_x(x_i, y_i, x_j, y_j) = A \exp\left(-\frac{(x_i - x_j)^2}{2\sigma_x^2} - \frac{(y_i - y_j)^2}{2\sigma_y^2}\right), \tag{1}$$

where $A = 1$ is an amplification factor, σ_x and σ_y are the length-scale factors governing the impact on each vehicle with different ranges. Then we can compute the conditional distribution of the velocity v_x at any position (x, y) within the ROI via:

$$\mu_{v_x} = K_x(x, y, \mathbf{X}, \mathbf{Y}) K_x^{-1}(\mathbf{X}, \mathbf{Y}, \mathbf{X}, \mathbf{Y}) \mathbf{V_X} \tag{2}$$

$$\sigma_{v_x}^2 = K_x(x, y, \mathbf{X}, \mathbf{Y}) K_x^{-1}(\mathbf{X}, \mathbf{Y}, \mathbf{X}, \mathbf{Y}) K_x(\mathbf{X}, \mathbf{Y}, x, y), \tag{3}$$

where $\mathbf{X} \in \mathbb{R}^N$ and $\mathbf{Y} \in \mathbb{R}^N$ denote the vector of positions of all surrounding vehicles at time t, $\mathbf{V_X} \in \mathbb{R}^N$ denotes the

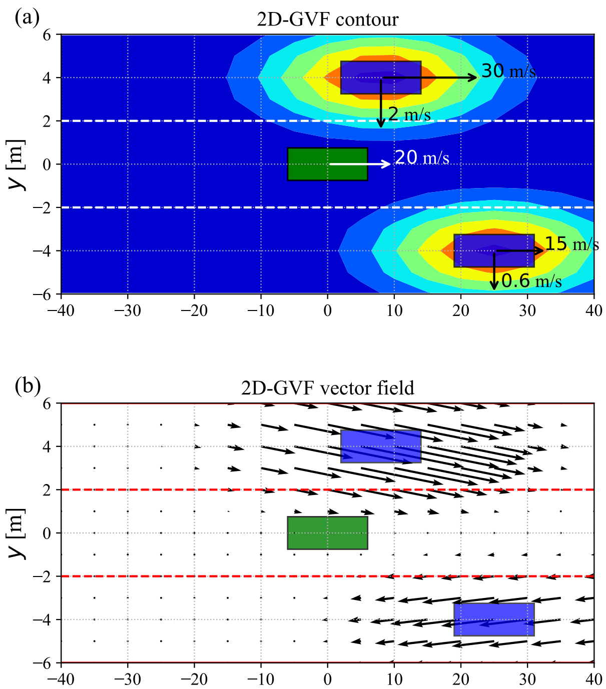


Fig. 2: The GVF in highway scenarios.

vector of $x-$direction velocities of all surrounding vehicles at time t. Similarly, the calculation of v_y is similar to the procedure above by using K_y:

$$K_y(x_i, y_i, x_j, y_j) = A' \exp\left(-\frac{(x_i - x_j)^2}{2\sigma_x'^2} - \frac{(y_i - y_j)^2}{2\sigma_y'^2}\right), \tag{4}$$

where we use the prime notation to denote the parameters using for computing the v_y, so A' is the amplification factor, σ_x' and σ_y' are the length-scale factors. Then, the conditional distribution of the velocity v_y can be derived accordingly:

$$\mu_{v_y} = K_y(x, y, \mathbf{X}, \mathbf{Y}) K_y^{-1}(\mathbf{X}, \mathbf{Y}, \mathbf{X}, \mathbf{Y}) \mathbf{V_Y}, \tag{5}$$

where $\mathbf{V_Y} \in \mathbb{R}^N$ denotes the vector of $y-$direction velocities of all surrounding vehicles at time t. Since only the expectation function is adopted in the construction of GVF, here we omit the covariance computations.

We illustrate the GVF of a specific highway environment, i.e., two obstacles with the relative velocities $v_1 - v_0$ and $v_2 - v_0$, in Fig.2 to help with a better understanding. In general, the dimension of GVF is similar to an RGB image, *i.e.*, $w \times h \times 2$, where 2 represents the components in two directions in x and y.

2) Improved Gaussian velocity field (3D-GVF): Given that 2D-GVF is constructed based on relative velocity, the amplitude may decrease and tend to 0 when the velocities of the target vehicle and surrounding obstacles are approaching to be the same. However, once the amplitude of 2D-GVF is approaching 0, it would be hard to determine whether the surrounding vehicle exists or not only given the sparse observations. To this end, it would be necessary to introduce an another dimension that indicate the existence of the surrounding vehicles. In this part, we will construct an improved Gaussian Velocity field with three dimensions, we denote as 3D-GVF.

To construct the 3D-GVF, similar to the previous procedures, we first define a standard squared exponential kernel similar as (1) and (4), we denote as K_I:

$$K_I(x_i, y_i, x_j, y_j) = A'' \exp\left(-\frac{(x_i - x_j)^2}{2\sigma_x''^2} - \frac{(y_i - y_j)^2}{2\sigma_y''^2}\right), \tag{6}$$

where we use the double-prime notation to denote the parameters using for computing the third dimension, so A'' is the amplification factor, σ_x'' and σ_y'' are the length-scale factors. The aim here is to estimate an conditional indicator function based on the observation data. Thus, we define a vector $\mathbf{I} \in \mathbb{R}^N$ whose elements are all equals to 1. By introducing this indicator, one can represents the existence of surrounding vehicles in the ROI following the typical method of 2D-GVF. Thus, the conditional distribution is computed as:

$$\mu_I = \alpha K_I(x, y, \mathbf{X}, \mathbf{Y}) K_I^{-1}(\mathbf{X}, \mathbf{Y}, \mathbf{X}, \mathbf{Y}) \mathbf{I}. \tag{7}$$

In general, the dimension of this 3D-GVF is similar to what it has in 2D-GVF but the channel number, *i.e.*, $w \times h \times 3$, where 3 represents the components of $x-$direction and $y-$direction velocities and an indicator.

B. Imitation Learning Models

The developed GVF captures the dynamic traffic environment using the estimated relative velocity at each location in the environment, formed as a tensor. Fortunately, the CNN model can handle these tensor very well. Here we establish a CNN model, named as GVF-CNN, to make adaptive decisions according to the observation in the traffic environment. The architecture of the GVF-CNN is shown in Table II, where the input is the GVF tensor with size $w \times h \times \mathbf{c}$, here $\mathbf{c}$ denotes the channel number. In the training of this model, the input data is the length, width and speed information of the velocity field. The output is one of the corresponding action in the action sets: (1) **turn left**, (2) **idle**, (3) **turn right**, (4) **go faster**, (5) **slow down**. Therefore, lane changing decisions of autonomous driving vehicles can be made.

IV. Experiments

In our experiment, we develop a motion planner that can make the vehicle navigate from the initial position x_s to the target position x_t while avoiding the obstacle x_o between them. Specifically, by defining x and y to denote position

TABLE II: GVF-CNN Architecture.

Layer	Kernel	Feature map	Padding
Input	–	$13*17*\mathbf{c}$	–
Conv1 + MaxPooling1	$3*3*64$	$10*14*64$	valid
Conv2 + MaxPooling2	$3*3*64$	$7*11*64$	valid
Conv3 + MaxPooling3	$2*2*128$	$6*10*128$	same
Conv4 + MaxPooling4	$2*2*128$	$5*9*128$	same
Conv5 + MaxPooling5	$2*2*128$	$3*7*128$	valid
Conv6 + MaxPooling6	$2*2*128$	$1*5*128$	valid
Flatten	–	640	–
FullyConnect1	–	216	–
FullyConnect2	–	128	–
Softmax	–	5	–

* Activation functions: 'ReLu'.

coordinates and θ to represent the orientation angle, the detailed settings of this approach are as follows:

- **Vehicle:** a vehicle which can ignore its shape and size. Except when detecting collisions in the simulator, vehicles are treated as mass points at other times. When detecting collisions, we treat a bird's-eye view vehicle as a rectangle and the coordinates of the lower left corner of the rectangle are recorded. In this way, we can calculate the positions of the four points of the rectangle to determine whether there is a collision.
- **Simulator:** *highway-env* [6]. It is a collection of environments for autonomous driving and tactical decision-making tasks which is typical.
- **Environment:** a single-direction congested highway.
- **Start position $\mathbf{x_s}$:** a random position on the lane.
- **Obstacle $\mathbf{x_o}$:** N surrounding vehicles ahead of the target vehicle.
- **Target position $\mathbf{x_t}$:** a position which is infinity far away, *e.g.*, 5000 m.
- **HighD dataset:** a new dataset [30] of naturalistic vehicle trajectories recorded on German highways. Traffic was recorded at six different locations and includes more than 110500 vehicles. Each vehicle's trajectory, including vehicle type, size and manoeuvres, is automatically extracted in this dataset which makes it suitable for the training of drivers' models.

A. Experiment Setup

1) highD dataset and pre-training: We use the lane-change data from the released Highway Drone (highD) Dataset [30] as the main training data. The highD dataset has 60 video recordings, logged with the sampling frequency of 25 Hz, consisting of two different environment settings: four-lane divided and six-lane divided. We selected one record from each setting – Recording03 and Recording11. To reduce the required computer memory, we downsampled the raw sequential frames to 5 Hz, thus obtaining 18281 frames of all lane-change vehicles. The parameters of establishing GVF are the same as in [26].

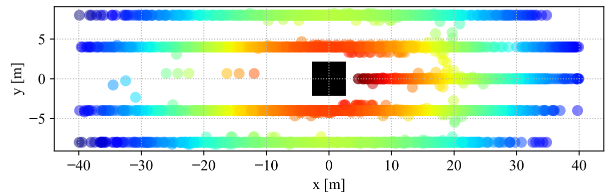


Fig. 3: The distribution of the closest obstacles at each frame around ego vehicle, right is the driving direction.

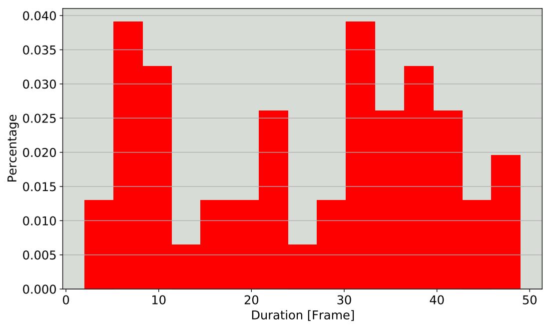
By training on the highD dataset, the GVF-CNN model is with 82.84% accuracy on the training set.

2) OPD data collection: Considering the differences between real-world data and simulation data, we collected some experimental data in the simulator by using a built-in algorithm, *i.e.*, OPD [7]. The average time of one step using OPD is approximately 2.36 s, which is extremely computationally expensive. Thus, due to limited time and computational resources, only 281 trials are conducted, which comes up with 13842 frames in total. We trained the GVF-CNN for 200 epochs with 75% frames for training set, 15% for validation set, and 10% for test set. Finally, it reaches 93.11% accuracy on the whole data.

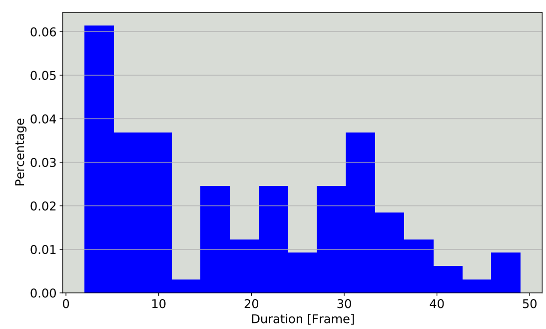
3) Kinematic bicycle model: Here we adopt the kinematic bicycle model [31] for ego vehicle and the intelligent driver model (IDM) [32] for other vehicles (obstacles). For ego vehicle, two low-level controllers are designed to control the vehicle: 1) longitudinal controller, which is a simple proportional controller; and 2) lateral controller, which is a simple proportional-derivative controller.

B. Results and Evaluation

1) Experiments with 2D-GVF: We run 500 rounds of simulations with 118 times crashed while 382 times survived, thus the crash rate is 23.6%. To evaluate the simulation results, we collected some data for analysis during the simulation. If the target vehicle crashed with surrounding vehicles, we collected the frames of duration. Otherwise, if the target vehicle survived through the congested highways, the distances to the closest vehicle at each frame is recorded. According to the data we collected, two histograms were drown to show the features clearly. We tried to find the pattern in the experiment results. Fig. 3 illustrates the position distribution of the closest vehicles in each frame. It is a heat map which the closer the color of the area is to red, the greater the probability of obstacles being distributed in this area; while the closer the color of the area is to blue, the smaller the probability of the obstacles being distributed in this area. As clearly shown in this heat map, the most closest area to the red color is right before the target vehicle of the driving direction, which is reasonable. As shown in Fig. 4a, it is the duration for failure cases. We can see there are two crests in the



(a)



(b)

Fig. 4: The histograms of duration frames for crashed experiments. (a) The experiment results with 2D-GVF. (b) The experiment results with 3D-GVF.

graph. One is at the beginning of the time line and the other one is in the middle. The reasons for these two crests are different, which we will talk about in the next part. And the red histogram in Fig. 5 shows the closest distances for survived cases. We can see that for those vehicles that did not crash, their closest distance is relatively small in value. It means our method is practical and explainable.

2) Experiments with 3D-GVF: As for the improved 3D-GVF model, We also run 500 rounds of simulations with 36 times crashed while 464 times survived, thus the crash rate is 7.2%. Compared with the crash rate of 2D-GVF model, the crash rate of 3D-GVF is much smaller. As shown in Table III, we can clearly see that the improved GVF has a better performance. Then we drew histograms the same way as experiments with 2D-GVF. As shown in Fig. 4a, it is the duration for failure cases. And the blue histogram Fig. 5 shows the closest distances for survived cases. To summarize, we pre-trained our GVF-CNN on 18281 frames from highD dataset, and fine-tuned the model using 13842 frames from OPD data collected by 281 rounds of experiments in simulator. The model reaches 93.11%

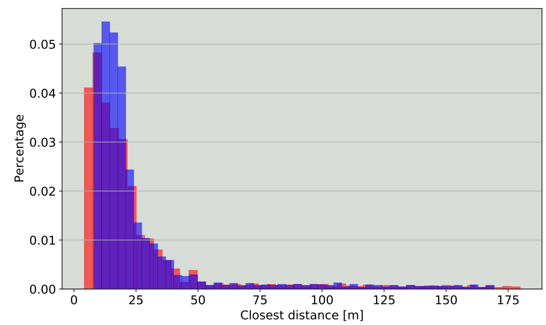


Fig. 5: The histogram of distances of the closest vehicle in each frame. The red histogram refers to closest distance with 2D-GVF while the blue one refers to closest distance with 3D-GVF.

TABLE III: Comparison of experiment results between 2D-GVF and 3D-GVF.

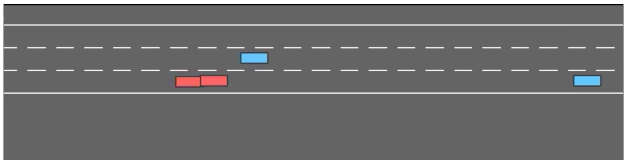
\	**2D-GVF**	**3D-GVF**
Rounds of simulations	500	500
Rounds which crashed	118	36
Rounds which survived	382	464
Crash rate	23.6%	7.2%

accuracy on the OPD data and our method also reaches high computation efficiency.

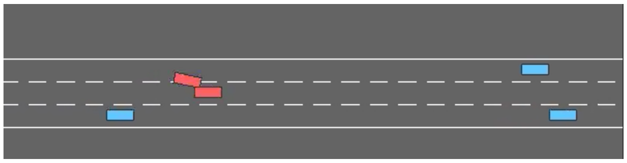
C. Analysis of Failure Cases

In this section, we will explicitly analyze how collisions happen. In general, mainly three reasons lead to the crash:

1) **a bad initial position:** as we randomly initialize the start positions of both the ego vehicle and obstacles, it may quickly lead to a crash results in these circumstances. As can be seen in Fig.4a and Fig.4b, the most left clusters with short duration can represent this case. However, this can be solved by additionally use certain number of frames as a pre-running period, during which the simulation data are not collected.
2) **rear-end collision:** as shown in Fig.6a, rear-end collision is also one of the most frequent failure cases. In such case, the ego vehicle would follow the obstacles closely, and trying many times between 'go faster' and 'slow down', but the relative velocity is approximately zero, which can hardly find the obstacles in the GVF. Thus, it is mainly due to the property of the feature we selected – not sensitive to the position but only focus on the velocity. This happened in experiments with 2D-GVF, and can be solved by adding a third channel $\mathbf{I}$ to represent the positions of obstacles. Therefore, with this kind of



(a)



(b)

Fig. 6: Two typical failure cases.

failure cases solved in experiments with 3D-GVF, the percentage of failure cases of the first kind increased comparing with that in experiments with 2D-GVF.

3) **passively collision:** as shown in Fig. 6b, sometimes it may happen that surrounding vehicles crash into the body of target vehicle. This may be solved by improving the model of surrounding vehicles in the simulator.

V. Conclusion

In this paper, we developed an imitation learning method to help autonomous vehicles make decisions when driving through congested highway scenarios. Firstly, we extracted a representative feature, named 2D-Gaussian velocity field which depicted the surrounding traffic environment. Using this feature makes the input data more uniform and there is no need to take the number of obstacles around target vehicles into consideration before training models. Secondly, we improved the 2D-Gaussian velocity field into three dimension and leveraged its accuracy. The 3D-GVF solved the problem that the value of 2D-GVF would disappear when the relative speed becoming smaller. Finally, we used the well-trained GVF-CNN model to predict high-level actions for the ego vehicles in the simulator and classified the actions into five sorts which helped make lane-changing decisions. With 500 rounds of simulations, we survived 464 times that implies the crash rate is 7.2%, and reasons of crashes are explicitly analyzed. In our future work, we will plan to integrate a continuous motion planner into the framework and evaluate our method in real life.

Acknowledgment

The research was financially supported by National Natural Science Foundation of China (No. 61972366), the Provincial Key Research and Development Program of Hubei (No. 2020BAB105), the Foundation of Henan Key Laboratory of Network Cryptography Technology (No. LNCT2020-A01), and the Foundation of Key Laboratory of Network Assessment Technology, Chinese Academy of Sciences (No. KFKT2019-003).

References

[1] D. Kasper, G. Weidl, T. Dang, G. Breuel, A. Tamke, A. Wedel, and W. Rosenstiel, "Object-oriented bayesian networks for detection of lane change maneuvers," *IEEE Intelligent Transportation Systems Magazine*, vol. 4, no. 3, pp. 19–31, 2012.

[2] J. Zhang and A. El Kamel, "Virtual traffic simulation with neural network learned mobility model," *Advances in Engineering Software*, vol. 115, pp. 103–111, 2018.

[3] V. Leonhardt and G. Wanielik, "Feature evaluation for lane change prediction based on driving situation and driver behavior," in *2017 20th International Conference on Information Fusion (Fusion)*. IEEE, 2017, pp. 1–7.

[4] Q. H. Do, H. Tehrani, S. Mita, M. Egawa, K. Muto, and K. Yoneda, "Human drivers based active-passive model for automated lane change," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 1, pp. 42–56, 2017.

[5] E. Galceran, A. G. Cunningham, R. M. Eustice, and E. Olson, "Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction: Theory and experiment," *Autonomous Robots*, vol. 41, no. 6, pp. 1367–1382, 2017.

[6] E. Leurent, "An environment for autonomous driving decision-making," https://github.com/eleurent/highway-env, 2018.

[7] J.-F. Hren and R. Munos, "Optimistic planning of deterministic systems," in *European Workshop on Reinforcement Learning*. Springer, 2008, pp. 151–164.

[8] Ü. Dogan, J. Edelbrunner, and I. Iossifidis, "Autonomous driving: A comparison of machine learning techniques by means of the prediction of lane change behavior," in *2011 IEEE International Conference on Robotics and Biomimetics*. IEEE, 2011, pp. 1837–1843.

[9] T. Li, Y. Chen, Y. Wang, Y. Wang, M. Zhao, H. Zhu, Y. Tian, X. Yu, and Y. Yang, "Rational protocols and attacks in blockchain system," *Security and Communication Networks*, vol. 2020, 2020.

[10] C. Vallon, Z. Ercan, A. Carvalho, and F. Borrelli, "A machine learning approach for personalized autonomous lane change initiation and control," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 1590–1595.

[11] M. Nazari, A. Oroojlooy, L. V. Snyder, and M. Takáč, "Reinforcement learning for solving the vehicle routing problem," *arXiv preprint arXiv:1802.04240*, 2018.

[12] G. Wang, J. Hu, Z. Li, and L. Li, "Cooperative lane changing via deep reinforcement learning," *arXiv preprint arXiv:1906.08662*, 2019.

[13] T. Li, Z. Wang, G. Yang, Y. Cui, Y. Chen, and X. Yu, "Semi-selfish mining based on hidden markov decision process," *International Journal of Intelligent Systems*, 2021.

[14] S. Yang, W. Wang, C. Lu, J. Gong, and J. Xi, "A time-efficient approach for decision-making style recognition in lane-changing behavior," *IEEE Transactions on Human-Machine Systems*, vol. 49, no. 6, pp. 579–588, 2019.

[15] D.-F. Xie, Z.-Z. Fang, B. Jia, and Z. He, "A data-driven lane-changing model based on deep learning," *Transportation research part C: emerging technologies*, vol. 106, pp. 41–60, 2019.

[16] M. G. Li, B. Jiang, Z. Che, X. Shi, M. Liu, Y. Meng, J. Ye, and Y. Liu, "Dbus: Human driving behavior understanding system," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE, 2019, pp. 2436–2444.

[17] H. Woo, Y. Ji, H. Kono, Y. Tamura, Y. Kuroda, T. Sugano, Y. Yamamoto, A. Yamashita, and H. Asama, "Lane-change detection based on vehicle-trajectory prediction," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1109–1116, 2017.

[18] V. Mahajan, C. Katrakazas, and C. Antoniou, "Prediction of lane-changing maneuvers with automatic labeling and deep learning," *Transportation research record*, vol. 2674, no. 7, pp. 336–347, 2020.

[19] Z. Wang, M. Guan, J. Lan, B. Yang, T. Kaizuka, J. Taki, and K. Nakano, "Analysis of truck driver behavior to design different lane change styles in automated driving," *arXiv preprint arXiv:2012.15164*, 2020.

[20] N. Deo, A. Rangesh, and M. M. Trivedi, "How would surround vehicles move? a unified framework for maneuver classification and motion prediction," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 129–140, 2018.

[21] H.-S. Jeon, D.-S. Kum, and W.-Y. Jeong, "Traffic scene prediction via deep learning: Introduction of multi-channel occupancy grid map as a scene representation," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1496–1501.

[22] S. Sivaraman and M. M. Trivedi, "Dynamic probabilistic drivability maps for lane change and merge driver assistance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2063–2073, 2014.

[23] L. Li, J. Gan, K. Zhou, X. Qu, and B. Ran, "A novel lane-changing model of connected and automated vehicles: Using the safety potential field theory," *Physica A: Statistical Mechanics and Its Applications*, vol. 559, p. 125039, 2020.

[24] K. Gao, D. Yan, F. Yang, J. Xie, L. Liu, R. Du, and N. Xiong, "Conditional artificial potential field-based autonomous vehicle safety control with interference of lane changing in mixed traffic scenario," *Sensors*, vol. 19, no. 19, p. 4199, 2019.

[25] C. Zhang, J. Zhu, W. Wang, and D. Zhao, "A general framework of learning multi-vehicle interaction patterns from video," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 4323–4328.

[26] C. Zhang, J. Zhu, W. Wang, and J. Xi, "Spatiotemporal learning of multivehicle interaction patterns in lane-change scenarios," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2021.

[27] W. Wang, C. Zhang, P. Wang, and C.-Y. Chan, "Learning representations for multi-vehicle spatiotemporal interactions with semi-stochastic potential fields," in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 1935–1940.

[28] X. Zhang, J. Sun, X. Qi, and J. Sun, "Simultaneous modeling of car-following and lane-changing behaviors using deep learning," *Transportation research part C: emerging technologies*, vol. 104, pp. 287–304, 2019.

[29] X. Liu, J. Liang, and B. Xu, "A deep learning method for lane changing situation assessment and decision making," *IEEE Access*, vol. 7, pp. 133 749–133 759, 2019.

[30] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2118–2125.

[31] P. Polack, F. Altché, B. d'Andréa Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 812–818.

[32] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical review E*, vol. 62, no. 2, p. 1805, 2000.