2019 IEEE Intelligent Transportation Systems Conference (ITSC)
Auckland, NZ, October 27-30, 2019

Human-like Highway Trajectory Modeling based on Inverse Reinforcement Learning

Ruoyu Sun, Shaochi Hu, Huijing Zhao, Mathieu Moze, Francois Aioun, Franck Guillemard

Abstract— **Autonomous driving is one of the current cutting edge technologies. For autonomous cars, their driving actions and trajectories should not only achieve autonomy and safety, but also obey human drivers' behavior patterns, when sharing the roads with other human drivers on the highway. Traditional methods, though robust and interpretable, demands much human labor in engineering the complex mapping from current driving situation to vehicle's future control. For newly developed deep-learning methods, though they can automatically learn such complex mapping from data and demands fewer humans' engineering, they mostly act like black-box, and are less interpretable. We proposed a new combined method based on inverse reinforcement learning to harness the advantages of both. Experimental validations on lane-change prediction and human-like trajectory planning show that the proposed method approximates the state-of-the-art performance in modeling human trajectories, and is both interpretable and data-driven.**

I. INTRODUCTION

In recent years, amazing progress has been demonstrated on the development of autonomous driving systems such as prototyping driver-less cars [1] and deep-learning based self-driving approaches [2]. However, a survey from the Univ. of Michigan‘s Transportation Research Institute shows that over 88.6% of the respondents show concerns about self-driving vehicles, while more than half respondents worried about whether self-driving vehicles can perform as well as actual drivers [3]. Autonomous driving agents should not only achieve autonomy and safety, but also follow human driver's behavior patterns when sharing roads with human drivers, so that their behaviors can be more predictable, and the passengers of an autonomous car and other traffic participants of the road may feel more comfortable.

Traditionally, this objective is considered in decision making and motion planning methods. Based on prior knowledge about human drivers, the designed methods firstly selects an appropriate high-level pattern for the vehicle (which could be, e.g., cruise-in-lane, change-lane, or turn-right), then it computes a safe, comfortable, and dynamically feasible trajectory from the vehicle's current configuration to future seconds [4]. However, previous approaches rely on hand-crafted modeling, and parameter tuning needs large amount of efforts by human experts, whereas the performance may not be satisfying in arbitrary complex real-world traffic environments. Recently, with the development of deep-learning methods, such objective is also covered in studies of learning an end-to-end driving agent [2], [5]–[10]. Using massive experts' driving data for supervision/imitation, the high-dimensional and highly non-linear mapping from sensory inputs of a driving situation (e.g. laser measurements or camera image) to its control outputs (e.g. steering angle and acceleration, or location coordinate) are directly learned. This is a great progress as it is much more convenient. However, as end-to-end mappings, such methods also skip the prediction of intermediate human-style driving patterns, such as humans' decision on car-following and lane-changing (literally studied as technical planning [11], [12]). This may cause the planned trajectory comparatively less human-like or interpretable. If the end-to-end driving agent make an error, passengers may have less chance to know and take action in time.

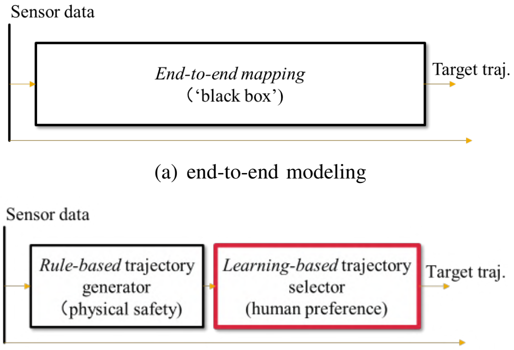
Inverse reinforcement learning (IRL) method learns a reward function from demonstration data [13] [14]. For trajectory planning, such a reward (negative cost) function is used to measure how 'good' a candidate trajectory is, then select the optimal one. The requirement to use IRL - the demonstration is satisfiable by recording expert drivers' trajectories. Recently, IRL method is used improve interpretability of 'deep' driving models' in urban [15], [16] and off-road [17] environments, by learn the cost maps (a 2-D reward function of vehicles' positions) which is visually understandable. However, current methods mainly tackle static environments. In dynamic traffic environment, a driving trajectory could be carried out by a number of decisions, and each is made based on the instantaneous observations at/up to the moment, which brings challenges.

Aiming at trajectory planning for autonomous driving at dynamic highway scenes, this research proposes a rule-based and learning-based combined methodology for human-like trajectory modeling, by exploiting maximum entropy inverse reinforcement learning [14]. It use two stages to take the advantages of both the robustness and physical safety from rule-based trajectory generator (where there have been established methodology), and the automation and well-harness of data from learning-based reward function, as shown in Fig. I. The proposed methods are also inspired by recent efforts on human-like trajectory planning [18], but using data-driven methods as IRL without hand-designing cost functions. It provides an appropriate combination between humans' prior knowledge (rule-based methods) and machine-learning (learning-based methods). Experimental results are used to validate the method's effectiveness in human driving prediction and trajectory planning, as well as its interpretable

This work is partially supported by Groupe PSA's OpenLab program (Multimodal Perception and Reasoning for Intelligent Vehicles) and the NSFC Grants 61573027. R.Sun, S.Hu, H.Zhao are with the Key Lab of Machine Perception, Peking University, Beijing, China. M.Moze, F.Aioun and F.Guillemard are with Groupe PSA, Velizy, France.

Contact: H.Zhao, zhaohj@cis.pku.edu.cn.

978-1-5386-7024-8/19/$31.00 ©2019 IEEE



(a) end-to-end modeling

(b) reward-based modeling

Fig. 1. Comparison of two kinds of trajectory modeling methods. Reward-based modeling is inclined in this research.

features. The specific structural design of the reward function is another worthy researching topic, but our study shows that, even with current vanilla implementation, the proposed methodology can already achieve promising results. e

This paper is organized as follows. A brief review of related work is given in Section II. The proposed methodology is described in Section III. Experimental validations are given in Section IV. Conclusion and future work will be addressed in Section V.

II. Related Work

To achieve human driver-like trajectory planning, traditional methods use decision making - motion planning - control framework. More specifically, a behavior-layer model is firstly responsible for making decisions on predefined high-level maneuver (which could be, e.g., cruise-in-lane, change-lane, or turn-right), then a motion planning module finds a feasible path or trajectory for it, which can be tracked by the low-level feedback controller [4], [19]. Such methods are based on experts' manual designs, have been widely used for reproducing traffic flow in simulation environment [20] due to their conciseness and robustness, and are fully surveyed for autonomous cars [4], [19], [21]. However, they may be bottlenecked by the limitation of human labor, in modeling accuracy of human driving behavior in future development.

With the recent development of deep-learning methods, deep neural networks are developed to directly learn an end-to-end mapping from the perceived information of a driver to its action [2], [5]–[8]. These researches achieve great progress as they can automatically learn to drive a car by taking advantages of massive data, with little manual engineering. However, they mainly learn to predict or generate real-time driving control (specifically, e.g., acceleration, steering control) instead of a visible trajectory. Such models act like a black-box to human, and make users hard to judge and avoid unexpected danger.

Trajectory prediction methods are alternative solutions for human drivers like trajectory planning, since predicting a seasoned driver's trajectory is equivalent to plan it. Their advantage is that a trajectory is understandable and 'known in advance'. Current methods are firstly predicting drivers' high-level maneuvers by classification, including cruise-in-lane, left-lane-change, right-lane-change, which could remain stable for a long driving period. As the maneuvers are determined, a long trajectory (e.g. Changes in positions for every 0.1 seconds in future 3 seconds) can be generated within it by regression [22]–[24]. The maneuvers can also be learned end-to-end from data [25]. However, it is still worthy to seek more scalable models, which can cover multiple maneuvers simultaneously.

Reward-based methods generally use a function to assess the gaining (or cost), and select the optimal action or trajectory from candidates according to it, instead of directly output them. Traditionally, such function is purely ruling-based, designed by humans, such as Potential Field method [26] and Rapidly-exploring Random Bees [27]. He et al. (2018) [18] learns the weight coefficient of a linear cost function, from humans' highway lane-change trajectories. It achieved great progress in selecting human-like trajectories from non-human-like ones, not just physical feasible trajectories. However, it still relies on manually engineering the features.

Inverse Reinforcement Learning (IRL) is a data-driven reward-based method, that can automatically learn the reward function. It is the inverse approach of Reinforcement Learning [28]. In RL, an optimal policy $p(a|s)$, which is the probability of choosing action a from situation s, is learned to maximize a known reward module $r(a, s)$. While in IRL, the reward model r is learned with known optimal policy [13]. The reward module can be not only a linear function [14] but also a deep neural networks [15]. The requirement of using inverse reinforcement learning, as the availability of optimal policy $p(a|s)$, can be satisfied by collecting data from some seasoned human drivers. [16] [17] use Deep IRL method to learn the cost map, which means the 2-D cost function of vehicle position, from humans' driving trajectory. The authors claim that by deploy such map, the trajectory similar to human drivers' will be selected from the candidates. Such methods work effectively with good interpretability in static environments, but may encounter unsolved challenges in highway environments, where moving surrounding vehicles exists. For example, there may be no cost map, as cost measurements will be determined by not only target vehicle's 2-D position, but also the time (so the 2-D cost function will no longer work).

III. Methodology

This research proposes a combined methodology for human-like trajectory modeling, used for human trajectory prediction and planning. It includes two parts. At first part, a set of feasible trajectories is first generated to using a rule-based method [29]. At the second part, a trajectory is then selected with the learned reward function, in compliance with human drivers' policies on the distribution of their demonstration data.

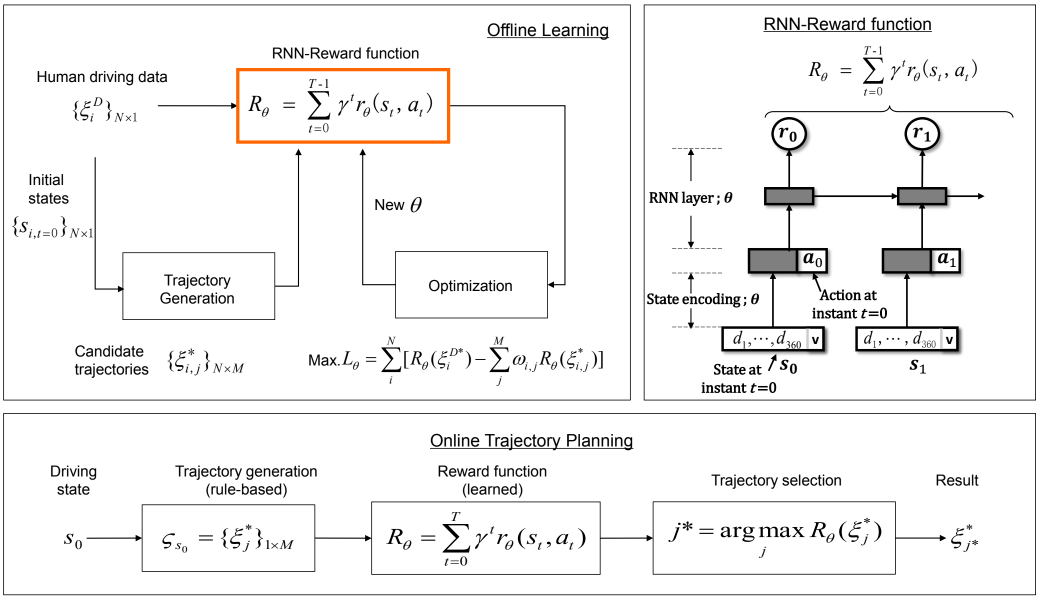

Offline Learning
RNN-Reward function
Human driving data
$\{\xi_i^D\}_{N\times 1}$
$R_\theta = \sum_{t=0}^{T-1} \gamma^t r_\theta(s_t, a_t)$
Initial states
$\{s_{i,t=0}\}_{N\times 1}$
New θ
Trajectory Generation
Optimization
Candidate trajectories
$\{\xi_{i,j}^*\}_{N\times M}$
Max. $L_\theta = \sum_i^N [R_\theta(\xi_i^{D^*}) - \sum_j^M \omega_{i,j} R_\theta(\xi_{i,j}^*)]$
RNN-Reward function
$R_\theta = \sum_{t=0}^{T-1} \gamma^t r_\theta(s_t, a_t)$
r_0
r_1
RNN layer ; θ
a_0
a_1
State encoding ; θ
Action at instant t=0
$d_1, \cdots, d_{360}$ v
$d_1, \cdots, d_{360}$ v
State at instant t=0
s_0
s_1
Online Trajectory Planning
Driving state
Trajectory generation (rule-based)
Reward function (learned)
Trajectory selection
Result
s_0
$\varsigma_{s_0} = \{\xi_j^*\}_{1\times M}$
$R_\theta = \sum_{t=0}^{T} \gamma^t r_\theta(s_t, a_t)$
$j^* = \arg\max_j R_\theta(\xi_j^*)$
$\xi_{j^*}^*$

Fig. 2. Methodology Outline.

A. Notation

The ith trajectory ξ_i in a set of demonstration samples $\{\xi_i\}_{1:N}$ can be represented as a sequence of T state-action pairs $\{(s_{i,0}, a_{i,0}), ..., (s_{i,T-1}, a_{i,T-1})\}$, where action $a_{i,t}$ are taken sequentially at situation $s_{i,t}$ at time t. For simplicity, we use s_i to denote $s_{i,0}$, which is the initial situation and the start state of each trajectory. Let $R_\theta(\xi, s)$ be the reward value of a trajectory $\xi = \xi_i$, which start from a certain initial situation $s = s_i$. Then the reward can be factorized as a discounted sum of the state rewards $r_\theta(a, s)$ along the time stream as below, with γ be a discount factor.

$$R_\theta(\xi, s) = \sum_{t=0}^{T-1} \gamma^t r_\theta(a_t, s_t) \tag{1}$$

Given a set of demonstration samples $D = \{\xi_i^D\}_{1:N}$ under a human driver's operation, our goal is to learn a reward function r_θ, which is used to choose the trajectories that are similar with human drivers behavior in trajectory planning.

B. Problem Formulation using Maximum Entropy IRL

Assume the probabilistic distribution $p(\xi|s_i)$ of all potential trajectories ξ at a initial driving situation s_i has the following structure.

$$\begin{aligned} p(\xi_i|s_i) &= \frac{\exp R_\theta(\xi_i, s_i)}{Z(\theta, s_i)} \\ Z(\theta, s_i) &= \int \exp R_\theta(\xi, s_i) d\xi \end{aligned} \tag{2}$$

Given a certain formulation of the reward function R_θ, learning a parameter set θ^* on N demonstration samples can be formulated as a maximum a posteriori estimation problem as below.

$$\begin{aligned} \theta^* &= \arg\max_\theta \prod_i^N p(\xi_i^D|s_i, \theta) \\ &= \arg\max_\theta \sum_i^N \log p(\xi_i^D|s_i, \theta) \\ &= \arg\max_\theta \sum_i^N [R_\theta(\xi_i^D, s_i) - \log Z(\theta, s_i)] \end{aligned} \tag{3}$$

C. Optimization

Off-line learning is to find the parameter θ^* of the reward function R_θ, defined by Eqn. 3, with human driving trajectories as demonstration samples. The general workflow is shown in left top subgraph in Fig. 2, and the details are given below.

Gradient ascend optimization is used to solve Eqn. 3. Let the right side of equation be defined as $L_\theta = \sum_i^N [R_\theta(\xi_i^D, s_i) - \log Z(\theta, s_i)]$, its derivative with respect to θ is given by

$$\frac{\partial L(\theta)}{\partial \theta} = \sum_i^N [\frac{\partial R_\theta(\xi_i^D, s_i)}{\partial \theta} - \frac{\partial \log Z(\theta, s_i)}{\partial \theta}] \tag{4}$$

The right item of the above equation can be deduced to

$$\begin{aligned} \frac{\partial \log Z(\theta, s_i)}{\partial \theta} &= \frac{\int \exp R_\theta(\xi, s_i) \frac{\partial R_\theta(\xi, s_i)}{\partial \theta} d\xi}{Z(\theta, s_i)} \\ &= \int p(\xi|s_i) \frac{\partial R_\theta(\xi, s_i)}{\partial \theta} d\xi \\ &= E_{\xi \sim p(\xi|s_i, \theta)} [\frac{\partial R_\theta(\xi, s_i)}{\partial \theta}] \end{aligned} \tag{5}$$

For each initial driving situation s_i in the demonstration samples D, a set of M trajectories $\zeta_{s_i} = \{\xi_{i,j}^*\}_{j=1,2,M}$

is generated to represent those candidates that a vehicle can drive through with the start point at s_i. Following the definition in Eqn.2, the probability of a candidate trajectory $\xi^*_{i,j}$ being selected for execution at situation s_i is estimated below.

$$\omega_{i,j} = P(\xi^*_{i,j}|s_i) = \frac{\exp R_\theta(\xi^*_{i,j}, s_i)}{Z(\theta, s_i)} \tag{6}$$

$$Z(\theta, s_i) = \sum_{m=1}^{M} \exp R_\theta(\xi^*_{i,m}, s_i)$$

Therefore, estimation of the expectation over all potential trajectories in Eqn.5 can be approximated by using trajectory sets ζ_{s_i}, and it is hence converted to the followings.

$$\begin{aligned} E_{\xi\sim p(\xi|s_i,\theta)}[\frac{\partial R_\theta(\xi, s_i)}{\partial\theta}] &\approx E_{\xi^*_{i,j}\sim\zeta_{s_i}}[\frac{\partial R_\theta(\xi^*_{i,j}, s_i)}{\partial\theta}] \\ &= \sum_{j=1}^{M}\omega_{i,j}\frac{\partial R_\theta(\xi^*_{i,j}, s_i)}{\partial\theta} \end{aligned} \tag{7}$$

For each situation s_i, we use the most similar (measured by distance, implementation can be found in Eqn. 12) trajectory to demonstration ξ^D_i from trajectory set ζ_{s_i}, denoted as $\xi^{D^*}_i$, to substitute ξ^D_i.

$$\frac{\partial R_\theta(\xi^D_i, s_i)}{\partial\theta} \approx \frac{\partial R_\theta(\xi^{D^*}_i, s_i)}{\partial\theta} \tag{8}$$

Substituting Eqn.7, Eqn.5 and Eqn.8 to Eqn.4, we have

$$\frac{\partial L(\theta)}{\partial\theta} = \sum_i^N[\frac{\partial R_\theta(\xi^{D^*}_i, s_i)}{\partial\theta} - \sum_j^M\omega_{i,j}\frac{\partial R_\theta(\xi^*_{i,j}, s_i)}{\partial\theta}] \tag{9}$$

By substitute Eqn.1 into Eqn.9, we have

$$\frac{\partial L(\theta)}{\partial\theta} = \sum_{i,t}^{N,T}\gamma^{t-1}(\frac{\partial r_\theta(a^D_{t,i}, s^D_{t,i})}{\partial\theta} - \sum_j^M\omega_{i,j}\frac{\partial r_\theta(a^*_{t,i,j}, s^*_{t,i,j})}{\partial\theta}) \tag{10}$$

which give details to the optimization shown in Fig. 2. It can be explained as an iterative process of maximizing the reward of demonstration trajectories, at the same time minimizing the weighted reward (the weight) of sample trajectories with respective initial situations.

D. Learning and Trajectory Planning

The processing flows of learning and trajectory planning are illustrated in Fig. 2. In offline learning, given a set of human driving data $\{\xi^D_i\}_{N\times1}$, sets of candidate trajectories $\{\xi^*_{i,j}\}_{N\times M}$ are generated based on the initial states $\{s_{i,t=0}\}_{N\times1}$. A reward function R_θ is learned to maximize the rewards of human driving trajectories, meanwhile minimize those of the candidate ones, i.e. maximize their difference.

The reward function R_θ is modeled in an RNN (recurrent neural network) structure, which is composed of two layers: state encoding and RNN, while θ refers to all trainable parameters. Action a_t is a 2-dim vector, consists of planned or real measured acceleration and yaw-rate of heading angle at time t. State s_t contains two parts: a descriptor of the scene obstacles $\wp_t$ and the ego vehicles speed v_t at the moment. As shown in Fig. 3, following the way of range scanning using a horizontal lidar sensor, we define $\wp = (d_1, ..., d_360, \dot{d}_1, ..., \dot{d}_360)$ at the ego frame, where d_k is the range distance of the nearest obstacle at angle k, while $\dot{d}_k$ is the obstacles velocity that is estimated by $d^t_k - d^{t-1}_k$. Hence, a state s_t is a 721-dim vector. It is fed into a single-layer perception, encoded into a 50-dim vector. After concatenating with a_t, the 52-dim vector is fed into RNN to estimate r_t of the state-action pair at the moment. The parameters of both single-layer perceptron at state encoding layer and RNN are trained along with the learning network.

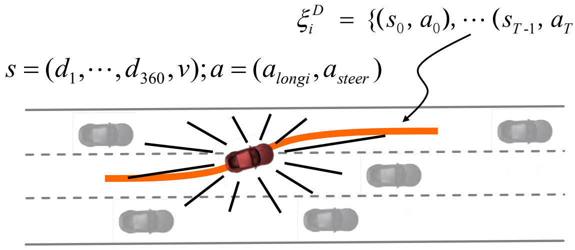

$\xi^D_i = \{(s_0, a_0), \cdots (s_{T\text{-}1}, a_{T\text{-}1})\}^D_i$
$s = (d_1, \cdots, d_{360}, v); a = (a_{longi}, a_{steer})$

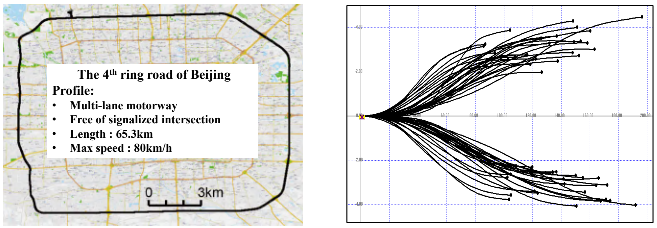
Fig. 3. State and action implementation.

Trajectory Planning is an online procedure as outlined in Fig. 2. Given initial driving state s_0 at the moment, a set of candidate trajectories $\zeta_{s_0} = \{\xi^*_j\}_{1\times M}$ is generated using rule-based method [29]. Assessing by the learned reward function, a trajectory is selected of the highest R_θ. It should be noticed that the future state $s^*_t (t > 0)$ of a candidate trajectory is unknown at the time of planning, which is synthesized using a linear motion model based on the driving situation at $t = 0$, i.e. s_0.

IV. EXPERIMENT

A. Data set

The data set [18] is used in this research, which was collected through on-road naturalistic driving on a multi-lane motorway in Beijing. We remove the lane change samples from the original data set that start from a side lane, to be sure that at the initial driving situation of all samples, the ego vehicle can choose to perform car following or change to either the left or the right lanes. As shown in Fig. 4, totally 175 lane-chang and 97 car-following samples are used in our experiment. Each sample contains the ego and surrounding vehicles trajectories during the period. Surrounding vehicles' location at each moment t is used to generate a scene descriptor s_t, while the ego vehicle's trajectory is used to recover the acceleration and yaw rates of a_t at each moment. 80 lane-change and 40 car-following samples are used for the test, and the others are used for training.



(a) (b)

Fig. 4. Data Collection Route and Lane Changing Trajectories

TABLE I
LANE CHANGE PREDICTION TRAINED BY 200 ITERATIONS

	RLC(P)	CF(P)	LLC(P)	Prediction Accuracy
RLC(G)	38	5	4	81%
CF(G)	13	23	4	56%
LLC(G)	7	2	24	73%

B. Implementing Details

1) Trajectory Generation: Given an initial driving situation s_0, a set of candidate trajectories are generated using Frenet framework [29]. Let $x(t)$ and $y(t)$ be the vehicle's longitudinal and lateral coordinate at time t with the origin at s_0, a trajectory can be represented as follow:

$$\begin{cases} x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \\ y(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5 \end{cases} \tag{11}$$

Where variables $a_0 - a_5$ and $b_0 - b_5$ can be determined, given the trajectory's start and end point's position, velocity, acceleration, and travel time during the course. In the experiment, for each human driven sample i, a trajectory set $\zeta^*_{s_i}$ is generated containing 10 left lane-change (LLC), 10 right lane-change (RLC), and one car-following (CF) ones. , that start from the initial velocity and position. An example of such a trajectory set is shown in Fig. 7, left.

2) Network and Training Configuration: In the experiment, trajectory points are tessellated by 0.1 second, i.e. interval between time t and $t+1$ is 0.1 seconds. We do not use the full course trajectory for learning and trajectory planning, but set T to 30. It means that the first 3 seconds of the trajectories are used in computing rewards. The RNN layer in Fig. 2 is implemented with 50 cells. The hyperparameter γ is set to 1. To train this neural network, we use the original gradient ascend method, with a learning rate of 0.1. We add a weight decay of 0.99 after each iteration. We use the full-batch training. The training process ends no less than 200 iterations, and no more than 400 iterations (each iteration contains one update of all trainable parameters).

TABLE II
LANE CHANGE PREDICTION TRAINED BY 400 ITERATIONS

	RLC(P)	CF(P)	LLC(P)	Prediction Accuracy
RLC(G)	32	10	5	68%
CF(G)	9	25	6	63%
LLC(G)	3	9	21	64%

C. Learning Results

This research aims at learning a reward function to evaluate the trajectory highest that is most probably selected by a human driver or most similar with human driven one. Therefore a distance measurement is defined below to evaluate the similarity of two trajectories, and relations between reward and trajectory distance are analyzed (the distance measurement is for validation, not used in training).

$$\sqrt{L2(\boldsymbol{v_1}, \boldsymbol{v_2})/\sigma(v) + L2(\boldsymbol{w_1}, \boldsymbol{w_2})/\sigma(w)} \tag{12}$$

where $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ means the velocity of the two trajectories in first 3 second segments. $\boldsymbol{w_1}$ and $\boldsymbol{w_2}$ means the direction change rate. $\sigma(v)$ and $\sigma(w)$ means the standard deviation of all velocity and change rate in the date. $L2$ means a 2-norm distance, and is computed as:

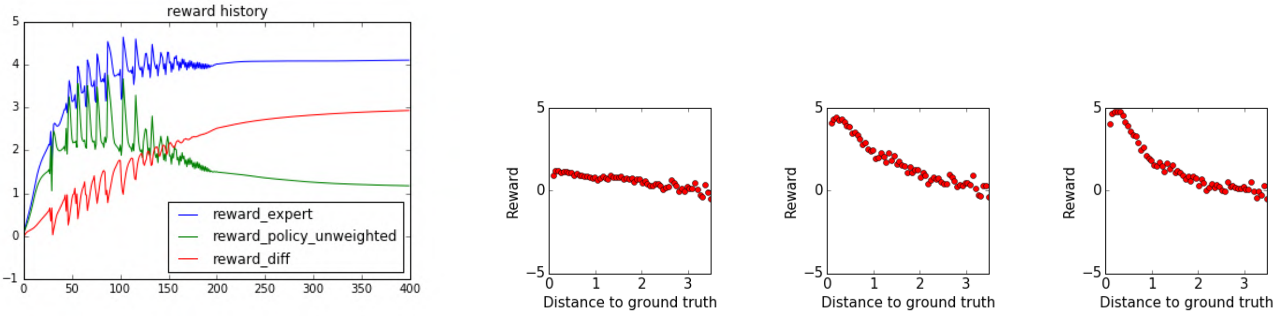
$$L2(\boldsymbol{a}, \boldsymbol{b}) = \sum(\boldsymbol{a} - \boldsymbol{b})^2 \tag{13}$$

As shown in Fig. 5(a), an average reward of the human drivers' trajectories (expert) is getting higher, while that of the candidates (policy) is getting lower. The difference between them is getting larger too with iteration times in learning. In Fig. 5(d), each is a result at the iteration of a certain human driven sample. A dot of each sub-figure is a candidate trajectory, cross-correlating reward with distance to the human driven trajectory (ground truth). It can be found that as iteration increasing, the tendency becomes clearer that the less distance a candidate trajectory (more similarity), the higher reward it has.

For each training or testing sample, the candidate trajectory that has the highest reward is selected, and the results are shown by the red bins in Fig. 6. The green line means selecting by ground truth (GT), which shows how far between the nearest candidates and the human expert driver's. Selecting by GT is the upper bound performance on current experimental setting. As can be seen, after the training above, by reward model $R_\theta(s, \xi)$, the selected trajectories mostly fall in the range of ground truth (GT). For the selections out of ground truth territory, they are still more similar to human driver's than un-trained random selection.

D. Lane Change Prediction Results

A similar experiment with [18] is conducted to examine the performance for lane change prediction. The results are shown in Table. I and Table. II. The integer number in each blank denotes the number of cases fell in the ground-truth(G), specified row-wise, and prediction (P), specified column-wise, category. Each accuracy for one of the three categories is computed as the ratio of, the number of cases that the prediction matches the ground-truth, to the number of all cases of that category. As can be seen, we get a prediction accuracy around 70% after 200 training iterations, and a prediction accuracy of 65% after 400 training iterations. The 200 iterations result has higher overall accuracy, but 400 iterations get a more balanced accuracy across three categories. Comparatively, this result above is lower than He



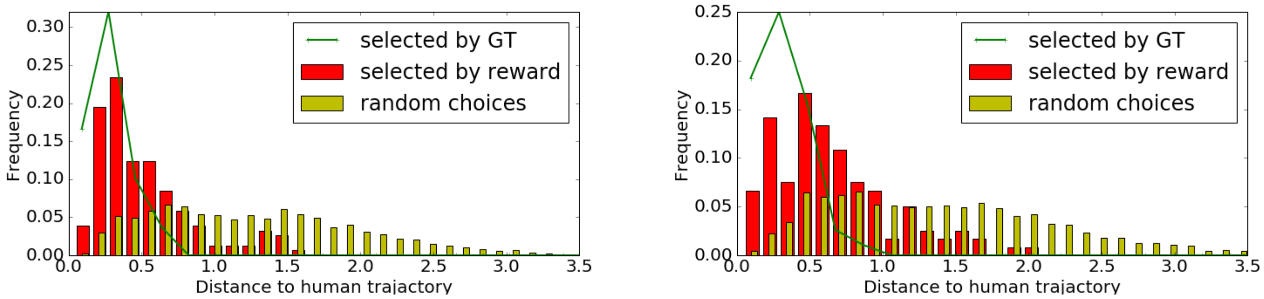
(a) The reward difference (red) of expert drivers' trajectories (blue) and all candidates (green) during training process.

(b) 20 iterations

(c) 100 iterations

(d) 200 iterations

Fig. 5. Training process visualization. The reward difference between expert drivers' and all candidate trajectories is shown on (a). Trajectories' reward distributions on how far it differ from ground truth, after 20 iterations, 100 iterations and 200 iterations are respectively shown on (b)-(d).



(a) Train set

(b) Test set

Fig. 6. The distribution of how far will the selected trajectory be different from seasoned human drivers' on (a) training set and (b) testing set. Less distance to human drivers' means better performance. Red bars mean selecting one trajectory per test case with highest reward, from all candidates. The yellow bars means selecting randomly for comparison. The green line means selecting the nearest candidates to human trajectory, which is the ground truth (GT) as described by Eqn. 8. It means reaching the upper bound performance if the distribution of 'selected by reward' completely fall in the range of the green line.

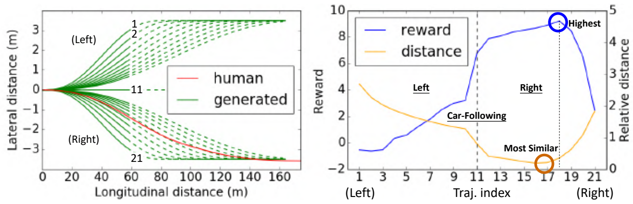


Fig. 7. One of the test cases. The left subgraph shows all the candidate trajectories, plotted as green lines, as well as the seasoned human driver's trajectory in the red line. Since only the first three seconds' segments are used as input, they are plotted as solid lines, and the segments beyond the first three seconds are plotted as dashed lines. The right subgraph shows the reward of each candidate trajectory, sequentially from left to right, as well as its relative distance of to the human drivers'. In this case, the experienced human driver's trajectory is a right lane-change, and by selecting according to the highest reward, we also get a similar right lane-change trajectory. So, this plan is human-like, and is a good result. More test cases are shown in Fig. 8,9,10 and 11.

et al. (2018) [18] which has a prediction accuracy around 74%. However, since the test settings are different, as we remove all cases that do not start changing lane from middle lanes (see Section IV.A) but He did not, and we engage less human engineering in designing features, achieve the same accuracy is more difficult. For example, in a case where the target vehicle drive on side lanes (they can only maintain current lane or change to one side), even a random choice will have an accuracy of 50%. But in all our test cases, a random choice only has an accuracy of 33% (and we got 65% - 70%).

E. Case by Case Show on Human-like Trajectory Planning

Each case is investigated to compare the planned trajectory with human drivers'. It also gives the visual explanation for the success and fail in the above lane change prediction experiment, and show the interpretability. An example case is shown in Fig. 7. The left subgraph shows all the candidate trajectories in green and the human driven one in red. As described in IV.B, the first 3 seconds of the trajectories are used in reward estimation, which are shown in solid lines The right subgraph plots the reward of each candidate trajectory, from index 1 to 21 in the same order with those of the left subgraph. In this case, the human driver made a right lane change, where the most similar candidate trajectory is #17, whereas #18 has the highest reward. With the learned reward function, correct lane change can be predicted (RLC), while the 2nd similar trajectory #18 is selected. Similar positive cases are shown in Fig. 8 and Fig. 9. Generally, there are

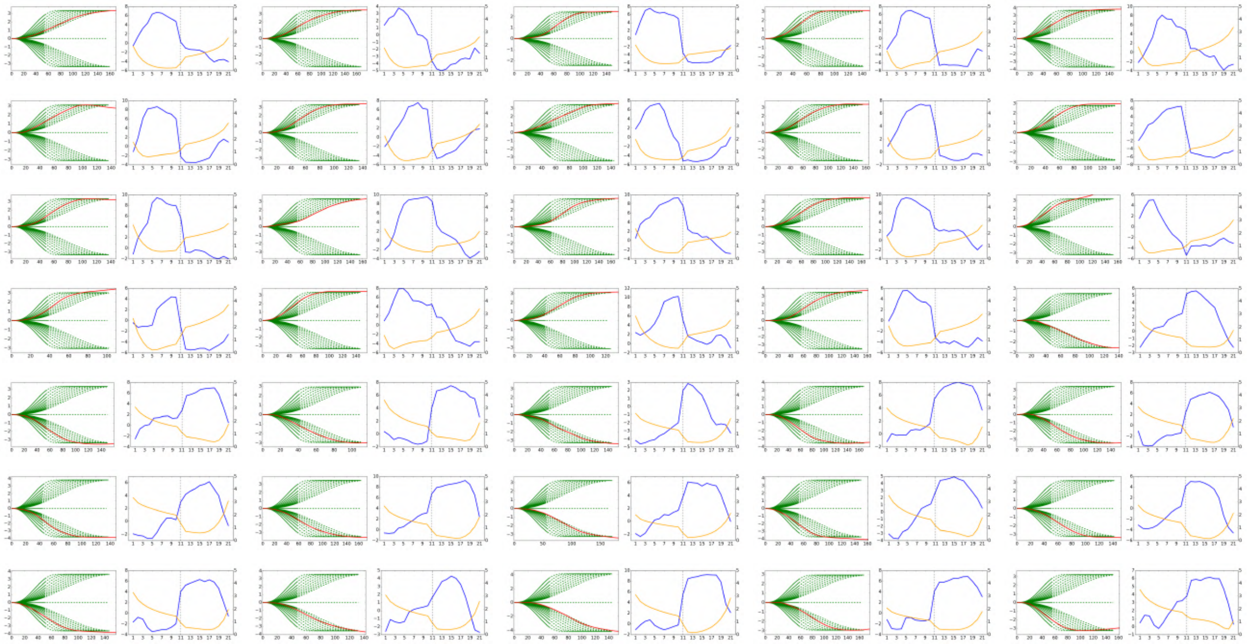


Fig. 8. Cases of human-like lane-change planning, and are good results. The pattern is that the trajectory reward (blue) is opposite to its distance to human driver's (yellow). This means that the planned trajectory has less distance and thus is similar to human drivers'. They corresponds to true lane change plan in TABLE.II. For example, if the human expert's trajectory (red line) is left lane-change, the planned trajectory, specifically the candidate trajectory with highest reward, is also in the same side. Due to limited page space, annotations can be found in Fig.7, and are omitted here.

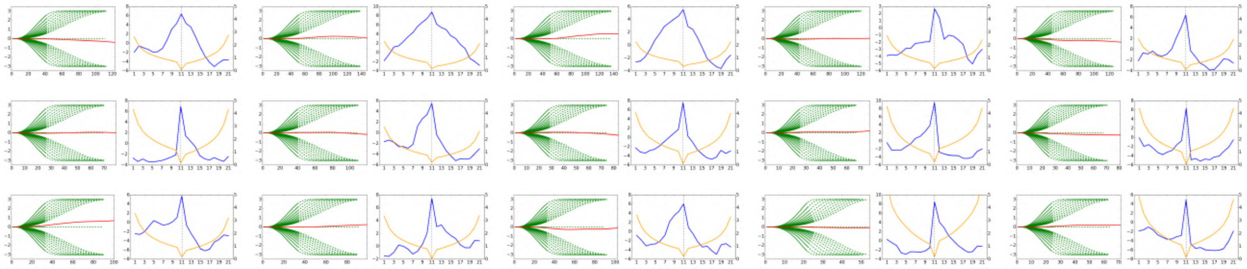


Fig. 9. Cases of human-like car-following planning, and are good results. In these cases, human driver plan to do car-following, and the car-following trajectory (the one in the middle) also has the highest estimate reward (plotted in blue).

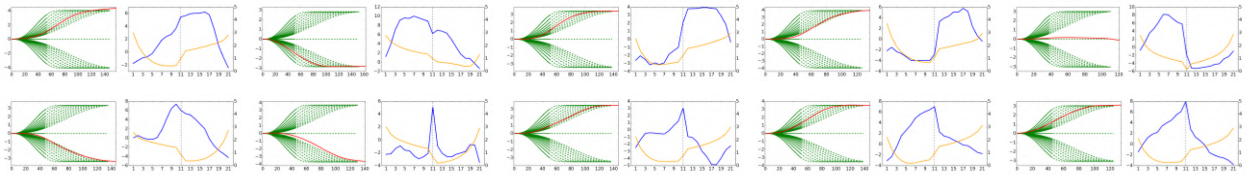


Fig. 10. Cases of none human-like planning, and are bad results. The pattern is that the candidate trajectories with highest reward (plotted in blue) has larger distance (plotted in yellow) to human driver's than most of the other. They correspond to false lane-change or car-following plans.

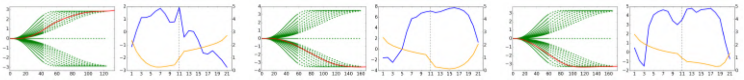


Fig. 11. Complex cases that it is unable to say whether they are good or bad, because a driver can only choose one trajectory to drive even if there are multiple choices. For example, in the first cases, maybe both left-lane change and car-following is possible, and the expert human driver choosing left lane-change does not necessarily mean the car-following one is bad. The third cases may correspond to situation of overtaking a front vehicle, and both left and right lane-change is viable, but the expert driver choose the right.

65%, which corresponds to TABLE. II. We also show some bad cases in Fig. 10. Besides, we find several complex cases shown in Fig. 11, where it may be hard to say whether they are good or bad, because a driver can only choose one trajectory to drive even if there are multiple equally good choices.

V. Conclusion and Future Work

In this research, we propose a method for learning a reward function from naturalistic driving data of human experts by exploiting maximum entropy inverse reinforcement learning (IRL). It combined following two parts. At trajectory planning, a set of feasible trajectories is first generated using a rule-based method. Then a trajectory is then selected with the learned reward function, which is in compliance with human drivers' policies on the distribution of their demonstration data. Experiments show promising performance in human driving prediction and planning, with good interpretability.

The structural design of the reward function is another interesting topic in the future. Currently, data samples needed for the experiments are very limited (e.g. naturalistic lane change trajectories), which form a generic restriction to test more complex neural network structures. The short of training data also easily causes false prediction, and forms the limitation in improving statistical performance in TABLE. II. More intensive and extensive examination of the proposed method that remains in future works as long as we collect more data. Also, as the trajectory generator is currently simple polynomial, we will validate whether a reward function trained with current trajectory generator can generalize when change to another trajectory generator.

References

[1] M. Dikmen and C. M. Burns, "Autonomous driving in the real world: Experiences with tesla autopilot and summon," in *Proceedings of the 8th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*. ACM, 2016, pp. 225–228.

[2] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[3] B. Schoettle and M. Sivak, "A survey of public opinion about autonomous and self-driving vehicles in the us, the uk, and australia," 2014.

[4] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on intelligent vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[5] J. Kim and J. Canny, "Interpretable learning for self-driving cars by visualizing causal attention," in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 2961–2969.

[6] L. Chi and Y. Mu, "Learning end-to-end autonomous steering model from spatial and temporal visual cues," in *Proceedings of the Workshop on Visual Analysis in Smart and Connected Communities*. ACM, 2017, pp. 9–16.

[7] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 3530–3538.

[8] Z. Yang, Y. Zhang, J. Yu, J. Cai, and J. Luo, "End-to-end multi-modal multi-task vehicle control for self-driving cars with visual perception," *arXiv preprint arXiv:1801.06734*, 2018.

[9] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, "Imitating driver behavior with generative adversarial networks," in *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE, 2017, pp. 204–211.

[10] N. Rhinehart, R. McAllister, and S. Levine, "Deep Imitative Models for Flexible Inference, Planning, and Control," Tech. Rep., 2018. [Online]. Available: http://arxiv.org/abs/1810.06544

[11] A. Doshi and M. M. Trivedi, "Tactical driver behavior prediction and intent inference: A review," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 1892–1897, 2011.

[12] T. Guy, J. M. Dolan, and J. W. Lee, "Automated tactical maneuver discovery, reasoning and trajectory planning for autonomous driving," *IEEE International Conference on Intelligent Robots and Systems*, vol. 2016-Novem, pp. 5474–5480, 2016.

[13] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 1.

[14] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning." in *AAAI*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[15] M. Wulfmeier, P. Ondruska, and I. Posner, "Deep inverse reinforcement learning," *CoRR, abs/1507.04888*, 2015.

[16] M. Wulfmeier, D. Z. Wang, and I. Posner, "Watch this: Scalable cost-function learning for path planning in urban environments," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 2089–2095.

[17] Y. Zhang, W. Wang, R. Bonatti, D. Maturana, and S. Scherer, "Integrating kinematics and environment context into deep inverse reinforcement learning for predicting off-road vehicle trajectories," *arXiv preprint arXiv:1810.07225*, 2018.

[18] X. He, D. Xu, H. Zhao, M. Moze, F. Aioun, and F. Guilemard, "A human-like trajectory planning method by learning from naturalistic driving data," *Intelligent Vehicles Symposium (IV)*, 2018.

[19] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles." *IEEE Trans. Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2016.

[20] J. Casas, J. L. Ferrer, D. Garcia, J. Perarnau, and A. Torday, "Traffic simulation with aimsun," in *Fundamentals of traffic simulation*. Springer, 2010, pp. 173–232.

[21] Z. Zheng, "Recent developments and research needs in modeling lane changing," *Transportation research part B: methodological*, vol. 60, pp. 16–32, 2014.

[22] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," *arXiv preprint arXiv:1805.06771*, 2018.

[23] W. Yao, H. Zhao, F. Davoine, and H. Zha, "Learning lane change trajectories from on-road driving data," in *Intelligent Vehicles Symposium (IV), 2012 IEEE*. IEEE, 2012, pp. 885–890.

[24] W. Yao, H. Zhao, P. Bonnifait, and H. Zha, "Lane change trajectory prediction by using recorded human driving data," in *Intelligent Vehicles Symposium (IV), 2013 IEEE*. IEEE, 2013, pp. 430–436.

[25] Y. Glassner, L. Gispan, A. Ayash, and T. F. Shohet, "Closing the gap towards end-to-end autonomous vehicle system," *arXiv preprint arXiv:1901.00114*, 2019.

[26] S. S. Ge and Y. J. Cui, "Dynamic motion planning for mobile robots using potential field method," *Autonomous robots*, vol. 13, no. 3, pp. 207–222, 2002.

[27] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2. IEEE, 2000, pp. 995–1001.

[28] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[29] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenet frame," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 987–993.