Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2016), pp. 1–11
Ladislav Kavan and Chris Wojtan (Editors)

A Data-driven Model for Lane-changing in Traffic Simulation

Huikun Bi[1,2,3], Tianlu Mao[2], Zhaoqi Wang[2], Zhigang Deng[3]

[1] the University of Chinese Academy of Sciences, Beijing, China
[2] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[3] Department of Computer Science, University of Houston, Houston, USA

Abstract

In this paper, we propose a new data-driven model to simulate the process of lane-changing in traffic simulation. Specifically, we first extract the features from surrounding vehicles that are relevant to the lane-changing of the subject vehicle. Then, we learn the lane-changing characteristics from the ground-truth vehicle trajectory data using randomized forest and back-propagation neural network algorithms. Our method can make the subject vehicle to take account of more gap options on the target lane to cut in as well as achieve more realistic lane-changing trajectories for the subject vehicle and the follower vehicle. Through many experiments and comparisons with selected state-of-the-art methods, we demonstrate that our approach can soundly outperform them in terms of the accuracy and quality of lane-changing simulation. Our model can be flexibly used together with a variety of existing car-following models to produce natural traffic animations in various virtual environments.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1. Introduction

In recent years traffic simulation has been widely used in urban planning, training, computer animation, computer games, and virtual reality [SBLM11, SWL11, WSL13]. In particular, with the increasing popularity of digital earths such as Google Maps, incorporating realistic traffic simulation into immersive urban environments has attracted noticeable attention.

In terms of simulating realistic traffic, existing traffic simulation techniques often fall into two categories: the automatic motion of a vehicle on a single lane (namely, car-following models) and the natural control of lane-changing among multiple lanes. These methods often focus on setting traffic rules or mechanisms to control individual vehicles; however, to date very limited efforts have been attempted on simulating realistic lane-changing behavior in traffic simulation.

In order to model complex driving behaviors such as natural lane-changing among multiple lanes, exploiting dynamic context factors during driving is necessary. For example, drivers typically need to take many factors into consideration before they commit lane-changing, including the minimal safety gap, the relative velocity of new leader vehicle, the relative velocity of new follower vehicle on the target lane, and so on. To date, how these factors would together affect the lane-changing behavior has been less studied in traffic simulation research. In this work, we argue that learning context-adaptive, lane-changing behavior from real-world traffic datasets is a sound strategy to tackle this issue, because real-world data implicitly encodes the complex correlations among these factors, which is advantageous over manually pre-coded rules.

Inspired by the above insight, in this paper, we present a new data-driven method to model context-adaptive, lane-changing behavior in traffic simulation. First, from a pre-collected vehicle trajectory dataset, we extract the features that are most relevant to lane-changing including its decision-making and execution processes. Second, based on the extracted features, we use machine learning algorithms to model both the lane-changing decision making process and the lane-changing execution process. Specifically, we train a randomized forest model for the lane-changing decision making process, and train a back-propagation neural network to estimate the vehicles' velocities involved in lane-changing. The velocities of new follower vehicles are also estimated based on the learned models.

We have conducted experiments to quantitatively and qualitatively evaluate the effectiveness of our approach, in particular, how our approach can generate realistic lane-changing behaviors in various driving environments. Furthermore, through comparisons with two selected state-of-the-art traffic simulation methods, we showed our method can soundly outperform them in terms of the accuracy and quality of lane-changing simulation.

The main contributions of this work can be summarized as follows.

- A new data-driven framework is designed to model natural lane-changing behavior in traffic simulation; it can produce more re-

alistic lane-changing simulation than the state-of-the-art, based on our experiments.
- Machine learning algorithms including both the random forests and back-propagation neural networks have been effectively adapted to model the high-level process (decision-making) and low-level process (execution process) of lane-changing behavior among multiple lanes in traffic simulation.

2. Related Work

Many traffic animation and simulation techniques have been proposed during the past several decades due to the ubiquity of vehicle traffic in our daily lives. In this section, we do not intend to comprehensively survey the developments in this field. Instead, we will only pay more attention to lane-changing models and existing efforts on applying machine learning algorithms for traffic simulation.

Generally, there are three types of traffic simulation techniques: *microscopic*, *mesoscopic*, and *macroscopic* methods. Microscopic simulation is also called agent-based methods that treats each vehicle as a discrete autonomous agent with pre-defined rules. Many agent-based methods use car-following rules to generate the behavior of vehicles in "stop-and-go" patterns [Ger55, New61, LCX*14]. Another well-known microscopic model is the Intelligent Driver Model (IDM) pioneered by Treiber et al. [THH00]. In addition, cellular automata was also applied to agent-based traffic simulation [NS92, TH02]. Mesoscopic methods use Boltzmann-type mesoscale equations to simulate traffic dynamics [PF60, SH99]. Finally, macroscopic models, also known as continuous methods, view traffic flow as continuum dynamics like fluid or gas. In these methods, researchers have used nonlinear scalar conservation law or other second-order systems of equations derived from the equations of gas dynamics to describe the regulations of vehicles [LW55, New61, Pay71, Whi11]. In this way, macroscopic models can deal with collections of vehicles and therefore work quite efficiently. Recently, simulations mixed of vehicles and pedestrians have also started to attract increasing attention [CDJ15].

Existing literature on lane-changing simulation is generally focused on two different aspects: modeling the decision-making process of lane-changing (that is, *when vehicles should change lanes*), and the execution of lane-changing (that is, *how vehicles perform lane-changing*). Hidas [Hid05] proposed to classify lane-changing behavior into free, forced and cooperative lane changes. In his work, lane-changing is simulated as an instantaneous action, and based on the minimal space gap, the lane-changing process with uniform accelerated motion is modeled as a function of speed. Later, researchers proposed a general model to derive lane-changing rules for discretionary and mandatory lane-changing for a variety of car-following models [KTH07]. Shen and Jin [SJ12] described a flexible continuous lane-changing model to safely simulate the driver's free or imperative lane-changing behaviors in order to overtake or other situations based on the work mentioned above. These works were designed to model the decision-making process for lane-changing, but they ignored the process of executing lane-changing as well as the velocities of other potentially involved vehicles. Recently, Sewall et al. proposed a hybrid traffic simulation framework, where the vehicles that attempt to change lanes obey the kinematic constraints [SWML10, SBLM11]. In this approach, vehicles are assumed to be traversable without abrupt wheel-steering, and the velocities of the vehicles are influenced by the curvature derivative of the lane-changing curve. The above previous approaches rely on pre-defined lane-changing rules, which ignores the context-adaptive characteristics of driving. Although these rules can describe and explain drivers' behaviors to certain extent, they often fall short of generating realistic lane-changing simulation due to the simplicity and inflexibility of such predefined rules. In addition, Mao et al. [MWDW15] proposed a general method to model lanes based on the road axis under the Frenet frame, by coupling mileage information with three-dimensional geometric information. It offers an easy and fast position transformation from mileage to the Cartesian coordinates in traffic simulation.

To tackle the limitations of rule-based methods, in recent years researchers have explored data-driven traffic simulation, that is, employing machine learning for traffic simulation. There is a large body of work in data-driven animation [MWLT13, KPAB15, LFCCO09]. Particularly, data-driven method also perform well in traffic simulation. For example, Meng and Weng [MW12] used a classification and regression tree (CART) approach, one of the most powerful data mining techniques to date, to predict the drivers' merging behavior in a work zone merging area. Hou et al. [HES14] applied Bayes classifier and decision-tree methods to model mandatory lane-changing at lane drops. Chong et al. [CAM11] proposed an agent-based neural network model to simulate the "stop-and-go" behavior in lanes. Later, they further developed a rule-based neural network model to simulate driver behavior in terms of longitudinal and lateral actions in two driving situations, namely, car-following situation and safety critical events [CAMH13]. Recently, Chao et al. [CSJ13] proposed a video-based approach with an offline learning process, in order to learn the specific driving characteristics of drivers for advanced traffic control. Despite these encouraging progresses of applying machine learning for traffic simulation, none of them has been focused on employing machine learning for generating realistic lane-changing simulation.

3. Our Method

As illustrated in Fig. 1, our method consists of three main modules: the offline preprocessing module, lane-changing decision making, and lane-changing execution. The *offline preprocessing* step extract most relevant features from a pre-collected traffic dataset after perform necessary data preprocessing. The *decision-making* module is designed to infer whether the subject vehicle should do lane-changing as well as which target lane/position it should change to. The *execution* module is designed to compute the detailed trajectories of the involved vehicles in order to accomplish the lane-changing task.

Decision-making for lane-changing: In terms of the decision-making for lane-changing, all the existing methods only judge the condition of the *adjacent* gap. The subject vehicle will do the lane-changing only If all of the gaps are sufficient, including the gap between the subject vehicle and the new leader vehicle, the gap between the subject vehicle and the new follower vehicle, and the velocity gaps of all the involved vehicles. Otherwise, the subject vehicle will keep waiting until all the conditions are met (refer to

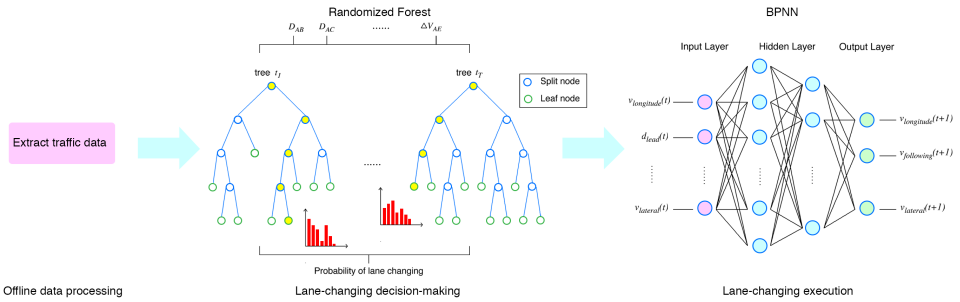


Figure 1: *Pipeline illustration of our data-driven lane-changing model*

Fig. 2(a)). However, under many real-world driving scenarios such as lane-merging and avoiding obstacles ahead, or drivers simply do not want to wait longer for lane-changing, vehicles often need to make prompt decisions on lane-changing.

Besides the *adjacent* gap on the target lane, our method also considers the *backward gap* and the *forward* gap on the target lane simultaneously, and attempts to complete lane-changing if any of the three gaps allows a safe lane-changing (illustrated in Fig. 2(b)). Now the question is *which of the three gaps a vehicle should choose for lane-changing at a particular moment*. To address this, based on a number of selected features including all the gaps and the relative velocities of all the directly involved vehicles, and the chosen gaps in real-world lane-changing traffic data, we train a randomized forest model to transform the gap selection problem into a classification problem. After the model training, we can automatically predict the probability of choosing a specific gap (i.e., the adjacent gap, the backward gap, or the forward gap) for lane-changing given a new traffic context.

Execution of lane-changing: At this step, we assume driving behavior can be conceptually regarded a state-action machine, where the driver's action at a specific moment heavily depends on his/her driving context including vehicle kinematic conditions and its surrounding environment. The states we consider in our approach are those features directly observed from traffic environment such as the vehicle velocity relative to the leader and follower vehicles on the target lane, and the leader and follower gap distances. Based on the ground-truth traffic data, we train a back-propagation neural network (BPNN) to encode the non-linear, complex mapping from the states of the involved vehicles to specific lane-changing actions.

Note that lane-changing can be typically classified into three types: free, forced, and cooperative lane-changes [Hid05]. Different from most of previous lane-changing works that were focused on *forced* lane-changing, our approach focuses on *cooperative* lane-changing, that is, simulating cooperative lane-changing among multiple lanes without collision.

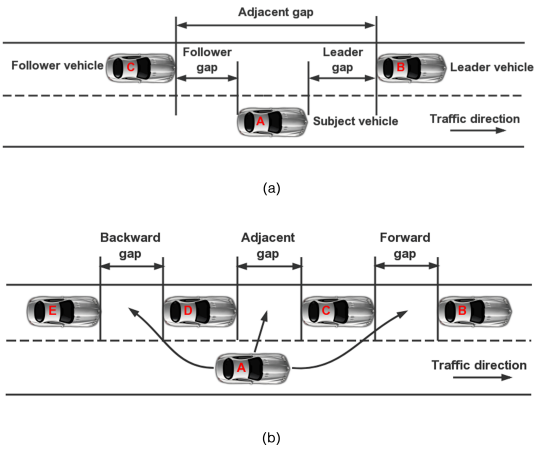


Figure 2: *Schematic differences between our method and previous lane-changing models: (a) the previous methods only focus on the adjacent gap on the target lane; (b) our method also considers the forward and backward gaps, besides the adjacent gap.*

4. Dataset

In this work, the used traffic trajectory data is from the Federal Highway Administration's Next Generation Simulation (NGSIM) dataset [ngs]. Some previous research studies showed that the NGSIM dataset exhibits certain noise such as random errors and measurement errors [PBC09, DBC09, KT08]. Therefore, we used Gaussian smoothing technique to process the accelerations, velocities, and coordinates of vehicles in the dataset.

First, we need to select those lane-changing samples involved with cooperation among vehicles. Specifically, we extract lane-changing samples based on the following criteria: (i) a lane-changing starts when a vehicle' lateral coordinate begins to shift toward the adjacent target lane direction without oscillations; (ii) similarly, a lane-changing ends when a vehicle keeps driving on the target lane without oscillations. Since our work is focused on

the cooperative lane-changing, the vehicles on the main lane and the adjacent target lane are of particular interest. When we selected the samples, we also ensured that only the subject vehicle changed lane and the order of the involved vehicles on the target lane (vehicles B, C, D, E in Fig. 2 (b)) were kept unchanged. Then, we need to extract the most relevant features for training our lane-changing model, including the accelerations, velocities, lateral and longitude coordinates of vehicles. Finally, the selected lane-changing samples were randomly divided into two groups: 80% for training and 20% for test purpose.

5. Decision-making for Lane-changing

In this work, we consider the following factors to affect the decision-making of lane-changing (illustrated in Fig. 3). These factors are used as input variables to our machine learning model.

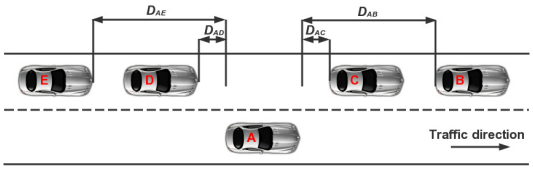


Figure 3: *Schematic illustration of the input variables used in the decision-making process of lane-changing in our approach*

- D_{AB} (or D_{AC}, D_{AD}, D_{AE}): The distance gap between the vehicle B (or C, D, E) on the target lane and the subject vehicle (A). D_{AB} can be expressed as

$$D_{AB} = D_B - D_A,$$

  where D_B is the location along with the traffic direction of the vehicle B, and D_A is the location along with the traffic direction of the subject vehicle (A). $D_{AB} < 0$ implies that the vehicle A is in front of the vehicle B along with the traffic direction.
- ΔV_{AB} (or ΔV_{AC}, ΔV_{AD}, ΔV_{AE}): The speed difference between the vehicle B (or C, D, E) on the target lane and the subject vehicle (A). ΔV_{AB} can be expressed as

$$\Delta V_{AB} = V_B - V_A,$$

  where V_B is the velocity of the vehicle B, and V_A is the velocity of the subject vehicle (A).

5.1. Methodology

To make a decision for lane-changing in this work, we need to learn a classification function F that takes $x = (D_{AB}, D_{AC}, D_{AD}, D_{AE}, \Delta V_{AB}, \Delta V_{AC}, \Delta V_{AD}, \Delta V_{AE})$ as the input and then predict its corresponding classification label y (the target gap that the subject vehicle will choose), based on the above collected lane-changing samples. Among many possible classification models, we choose randomized forests for our purpose, because the randomized forest method has been proven to be fast and effective for multi-class classification tasks [Cri11]. In the following, for the sake of completeness and readability, we will briefly describe the randomized forest method. Readers can refer to the work of [Cri11] for more algorithm details.

Basically, a decision forest is an ensemble of T decision trees. Each node n in a decision tree is associated with a learned class distribution $P(y|n)$. A decision tree works by recursively branching left or right down the tree according to a learned binary function of the feature vector, until a leaf node l is reached.

In the randomized learning process, each tree is trained separately on a random subset $I' \subseteq I$, where I denotes the training data. When we generate the i-th decision tree during training, it will randomly select m factors mentioned above as features. The goal of node splitting is to maximize the information gain. Here we adopt the classical Shannon's information theory to describe the information gain, described below.

For each feature (from 1 to m), each node n have three classes. Let y_1, y_2 and y_3 denote the three classes. We use $p(y_i|n)$ denote the probability of a sample in subset I_n belonging to y_i ($i = 1, 2, 3$). Then, the impurity of node n can be computed as:

$$I(n) = -\sum_{i=1}^{3} P(y_i|n) log_2 P(y_i|n)$$

$P(y_i|n)$ can be straightforwardly estimated by N_n^i/N_n, where N_n^i is the number of samples in I_n that belong to class y_i, and N_n is the total number of samples in I_n. Then, the information gain $\Delta I(n)$ that current feature would bring can be computed as:

$$\Delta I(n) = I(n) - \sum_{i=1}^{3} \frac{N_n^i}{N_n} I(y_i)$$

Among all the candidates, the one that can lead to the maximal decreasing of impurity is chosen. If the probability of samples in the subset belonging to any single class is more than P_0 (a predefined threshold), the splitting process will stop. Once the leaf node is reached, the class label is given by y_j, where

$$j = \arg\max_i P(y_i|n) \tag{1}$$

In our method, T = 20, and $P_0 = 0.9$. When we generate the i-th decision tree during training, we set the number of randomly selected features, $m < 4$.

After the model is trained, given a new set of input variables, we can automatically calculate the probability of choosing a specific target gap for lane-changing. In the lane-changing execution process, the subject vehicle will move to the target gap having the maximal estimated probability. If the estimated probabilities of all the three target gaps by our model are below a threshold, the subject vehicle will continue driving straightly and wait for a next appropriate moment for lane-changing.

6. Lane-changing Execution

In the lane-changing execution process, the subject vehicle needs to smoothly change from the current lane to the target gap that has

been selected in the above decision-making process, without causing any vehicle collisions. The subject vehicle need to adjust its velocity in both the lateral and longitude directions, based on all the factors observed in the surrounding environment. In this process, the velocity of the subject vehicle is calculated by a trained back-propagation neural network. Also, the new follower vehicle on the target lane would need to adjust its velocity to maintain an appropriate safe gap to avoid potential collisions.

6.1. Inputs of Lane-changing Execution

We keep the relative distance and velocity from the subject vehicle to the new leader vehicle, the relative distance and velocity from the subject vehicle to the new follower vehicle as a part of inputs to the employed machine learning model. Meanwhile, lane-changing execution is a continuous process; we also need to use the previous states of the involved vehicles as input variables to ensure the smoothness of vehicle movement. In addition, since the lane-changing behavior contains movement in the lateral direction, we also include the lateral velocity of the subject vehicle and the relative distance in the lateral direction from the current lane to the target lane as additional input variables. These input variables are detailed as follows (refer to Fig. 4 for illustration).

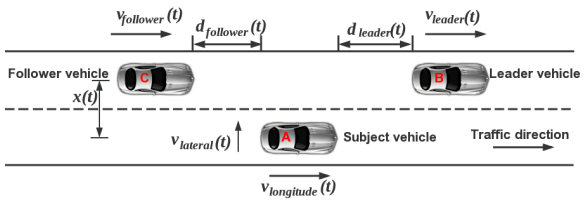


Figure 4: *Schematic illustration of the input variables to the learned lane-changing execution model*

- $v_{longitude}(t)$: The velocity of the subject vehicle in the traffic direction during lane-changing at t.
- $d_{leader}(t)$: The gap distance between the leader vehicle on the target lane and the subject vehicle in the longitude direction at t,

$$d_{leader}(t) = y_{leader}(t) - y(t),$$

where $y_{leader}(t)$ denotes the coordinate of the leader vehicle along the traffic direction, and $y(t)$ denotes the coordinate in the longitude direction.
- $d_{follower}(t)$: The gap distance between the follower vehicle on the target lane and the subject vehicle in the longitude direction at t,

$$d_{follower}(t) = y_{follower}(t) - y(t),$$

where $y_{follower}(t)$ denotes the coordinate of the leader vehicle along the traffic direction, and $y(t)$ denotes the coordinate in the longitude direction, the minus of $d_{leader}(t)$ and $d_{follower}(t)$ only represent the relative relationship between the vehicles.
- $\Delta v_{leader}(t)$: The velocity difference between the leader vehicle on the target lane and the subject vehicle in the traffic direction. $\Delta v_{leader}(t)$ can be expressed as:

$$\Delta v_{leader}(t) = v_{leader}(t) - v_{longitude}(t),$$

where $v_{leader}(t)$ is the velocity of the leader vehicle.
- $\Delta v_{follower}(t)$: The velocity difference between the follower vehicle on the target lane and the subject vehicle in the traffic direction. $\Delta v_{follower}(t)$ can be expressed as:

$$\Delta v_{follower}(t) = v_{follower}(t) - v_{longitude}(t),$$

where $v_{follower}(t)$ is the velocity of the follower vehicle.
- $\Delta x(t)$: The lateral distance to the target lane of the subject vehicle at t;
- $v_{lateral}(t)$: The velocity of the subject vehicle in the lateral direction during lane-changing at t;

6.2. Outputs of Lane-changing Execution

In order to take account of the cooperative behavior between the subject vehicle and the follower vehicle during lane-changing without collisions, our model need to calculate the velocities of the subject vehicle and the follower vehicle at any moment simultaneously. Specifically, the following three outputs are generated by our trained lane-changing execution model.

- $v_{longitude}(t+1)$: The velocity of the subject vehicle in the traffic direction during lane-changing at $t+1$;
- $v_{follower}(t+1)$: The velocity of the follower vehicle in the traffic direction when the subject vehicle changes its lane at $t+1$;
- $v_{lateral}(t+1)$: The velocity of the subject vehicle in the lateral direction during lane-changing at $t+1$;

To the end, the states of lane-changing execution, denoted by a vector, can be described as a function of the above mentioned outputs as follows.

$$\begin{aligned}&(v_{longitude}(t+1), v_{follower}(t+1), v_{lateral}(t+1))\\ &= f(v_{longitude}(t), d_{leader}(t), d_{follower}(t), \Delta v_{leader}(t),\\ &\qquad \Delta v_{follower}(t)), \Delta x(t), v_{lateral}(t))\end{aligned}$$

6.3. Methodology

In this work, we employ back-propagation neural network (BPNN) to model the lane-changing execution process. The BP learning algorithm we use can be divided into two phases: propagation, and weights update. At the propagation phase, it forward transfers training input through neural networks to generate the propagation's output activations. After that, the output activations are transferred through the neural networks using the target output to generate the gradients of all the output and hidden neurons. At the second phase (the weights update phase), the output delta and input activations are multiplied to obtain the gradients of the weights. Weights are brought in the opposite direction of the gradients by subtracting a ratio (the weight learning rate).

In one iteration, the outputs of BPNN can be calculated as follows:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_l \end{bmatrix} = \begin{bmatrix} h(\sum_{i=1}^{n_k} w_{i,1}^{k-1} s_{i,k-1} + b_1^k) \\ \vdots \\ h(\sum_{i=1}^{n_k} w_{i,l}^{k-1} s_{i,k-1} + b_l^k) \end{bmatrix}$$

where $h(\cdot)$ denotes the transfer function, $w_{i,l}$ is the weight connecting the i-th neuron in the $(k-1)$-th layer and the l-th neuron in k-th layer, $s_{i,k-1}$ is the value of the i-th neuron in the $(k-1)$-th layer, and b_l^k denotes the bias for the l-th neuron in the k-th layer. In addition, the BPNN learning in one iteration can be written as:

$$W_{k+1} = W_k - \alpha_k g_k,$$

where W_k is the vector of current weights and biases, α_k denotes the learning rate, and g_k denotes the current gradient. More details of BPNN learning can be found in the work of [HN89].

In our implementation, the used BPNN has 3 hidden layers, and each hidden layer has 5 nodes. A non-linear sigmoid transfer function is used as follows:

$$h(z) = \frac{2}{1+e^{-2z}} - 1$$

We set the learning rate $\alpha_k = 0.01$. At the 1st iteration, weights and biases are initialized as random values between 0 and 1.

In the lane-changing execution process, we take the traffic environment information of the subject vehicle as the inputs of the BPNN model. The trained BPNN can map the traffic environment information and the states of vehicles to the velocities of the subject vehicle and the follower vehicle at next moment during lane-changing.

6.4. Control of Boundary Conditions

Despite the above advantages of BPNN, it does not contain any physical significances. Therefore, we need to add boundary conditions to control the outputs of neural networks, namely, the velocities of the subject vehicle and the follower vehicle, as follows.

- *Maximum acceleration*: During the lane-changing process, the velocities of the subject vehicle and the follower vehicle are variational at each frame of simulation. The velocity we learnt also should obey the maximum acceleration of traffic data. If the velocity difference between two adjacent frames is more than the maximum acceleration, the vehicle will accelerate with the maximum acceleration.
- *Maximum deceleration*: Similarly, if the vehicles involved in lane-changing need to decelerate to avoid collisions, and the velocity difference we calculate through BPNN is more than the maximum deceleration, we simply set it to the maximum deceleration.
- *Minimum safe distance*: Due to the driving safety reason, the vehicles during lane-changing need to keep a minimal safe distance from the leader vehicle.

7. Experimental Results and Evaluations

We have implemented and tested our method on an off-the-shelf PC equipped with Intel Core (TM)2 CPU 6320@1.86 GHz, 8 GB main memory, and NVIDIA GetForce 8800 GTS graphics card.

7.1. Quantitative Analysis of Lane-changing Decision Making

Recalled in the above Section 5, we transform the lane-changing decision making problem (i.e., which gap on the target lane the subject vehicle should choose to switch to) to a classification problem. The used traffic data was 443 lane-changing samples selected from the NGSIM dataset. Among the selected samples, the subject vehicle chooses the forward gap in 45 samples (7277 frames), the adjacent gap in 233 samples (37244 frames), and the backward gap in 124 samples (20645 frames). We divided the above samples into two groups: 80% for training and 20% for test.

We also compared the randomized forest method with other well-known classification approaches including SVM (Support Vector Machine), GBDT (Gradient Boost Decision Tree), and Naive Bayes. Table 1 shows the obtained accuracies and the used computational time of all the methods on our test computer. Clearly, the results in Table 1 show that the randomized forest chosen in our method can soundly outperform the other methods in terms of accuracy.

Table 1: *Comparisons between the randomized forests with other well-known classifiers (SVM, GBDT, and Naive Bayes).*

Classifier	Accuracy	Used Time
Randomized Forest	91.03%	0.4838s
GBDT	86.2%	1.4270s
SVM	84.5%	5.1806s
Naive Bayes	85.62%	0.1438s

7.2. Quantitative Analysis of Lane-changing Execution

For different lane-changing cases (i.e., choosing the adjacent gap, forward gap, or backward gap), we trained different BPNN models. Without the loss of generality, in this section we present in details the quantitative analysis of the BPNN-based, lane-changing execution model for one specific case: when the subject vehicle chooses the adjacent gap as the target gap. We still used 80% of the data for training and 20% for test.

We also compared our method (BPNN-based) with a baseline method (that is, Radial Basis Functions (RBF)-based method) in Table 2. We used the R-squared to measure the simulated result. Then, we computed the percentage of test samples whose R-squared are in the range of [90%, 100%]. As shown in Table 2, our BPNN-based method can outperform the RBF-based baseline method in terms of the accuracy of lane-changing execution.

Table 2: *Comparison of our BPNN-based method and the RBF-based baseline method in terms of the accuracy of lane-changing execution.*

Learning Method	Accuracy	Time Used for Training (second)
BPNN	95.74%	179.38
RBF	88.12%	392.7

Figure 5 plots the resulting error distribution of our approach when it was applied to 47 test samples. We use the R-squared to

measure the simulated results of all the test samples. As illustrated in this figure, the R-squared of all the test samples is in the range of $[80\%, 100\%]$. We also can see that, among the 47 tested samples, the R-squared of the velocities-in-traffic-direction of the subject vehicle in 45 samples is more than 95%. In addition, the R-squared of the velocities-in-traffic-direction of the follower vehicle in 34 samples and that of the velocities-in-lateral-direction of the subject vehicle in 33 samples are more than 95%. These results validated that the simulated lane-changing execution of the vehicles by our method well approximate their real-world counterparts.

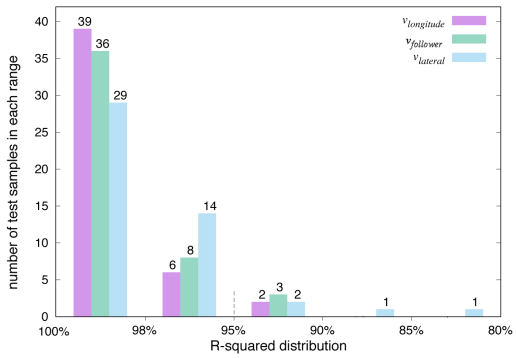


Figure 5: *Resulting error distribution of our approach when it was applied to 47 test samples.*

7.3. Simulation Results

In order to further test the performance of our method, we used our approach to simulate the *continuous* driving behaviors of the subject vehicle and the follower vehicle in lane-changing situations. Specifically, given only the trajectory data of the leader vehicle and the initiate states of the subject vehicle and the follower vehicle as well as the states of vehicles on the target lane, our method was used to generate continuous lane-changing movements automatically. These experiments are quite challenging, because the traffic states of the current frame is used as the inputs to the computation of next frame in this simulation process, which leads to accumulated simulation errors over frames.

Without loss of generality, we took the lane-changing simulation experiments for the vehicle #2719 (abbreviated as V2719) and the vehicle #2823 (abbreviated as V2823) as two examples (refer to Fig. 6). These two vehicles with quite different lane-changing patterns are picked randomly, but our method performed well for both cases. The velocity of the V2719 increases steadily and only decreases slightly at the last part of the lane-changing process. By contrast, the velocity of V2823 keeps somewhat stable at the first part, then declines quickly, and then increases steadily. Fig. 6 shows the comparisons among the ground-truth, the simulation results by our approach, and the simulation results by two selected state-of-the-art methods [SBLM11, SJ12].

As shown in Fig. 6(a) and (d), the simulated velocities in the longitude direction (that is, along with the traffic direction) of the subject vehicle are roughly close to the ground-truth, and perform better than the two state-of-the-art methods in terms of the overall pattern matching. Also, compared with the ground-truth, our method can produce more similar lane-changing patterns than the two selected methods, in terms of the velocities in the lateral direction (Fig. 6 (c) and (f)) and the velocities of the follower vehicle (Fig. 6 (b) and (e)). It is noteworthy that the velocity of the follower vehicle, computed by our method, slows down to make a larger gap for cutting if the current gap is too small for the subject vehicle (V2832), which is consistent with previous research studies in [Hid05]. However, if the current gap between the follower vehicle and the subject vehicle is sufficiently safe, the follower vehicle just drives mostly based on the its individualized driving states (V2719).

7.4. Comparison with State-of-the-art Methods

We also compared our method with two selected state-of-the-art methods [SBLM11, SJ12] in terms of lane-changing simulation. In terms of decision-making for lane-changing, both the above methods only check whether the adjacent gap on the target lane is available. Otherwise, they continue to repeat the checking process at the next moment. We randomly extracted 100 samples to test the decision-making results at the starting moment of lane-changing by different methods. As shown in Table 3, 25 vehicles chose the backward gaps for lane-changing, 50 vehicles chose the adjacent gaps, and the remaining 25 vehicles chose the forward gaps. Therefore, as indicated in Table 3, our method has substantially more chances to complete lane-changing without collisions than the two existing methods [SBLM11, SJ12]. The reason why the numbers for existing methods choose adjacent gaps are lower is that they use a curve to execute lane-changing and ingore the cooperation among vehicles.

Table 3: *Decision-making result comparison between our method and the two existing methods [SBLM11, SJ12].*

	Backward gaps	Adjacent gaps	Forward gaps
ground truth traffic	25	50	25
our method	13	44	21
Sewall et al.'s method	0	24	0
Shen and Jin's method	0	37	0

Fig. 6 compares some key variables in the execution of lane-changing by different methods in two test examples. As shown in Fig. 6(a) and (d), $v_{longitude}$ of the subject vehicle by Sewall et al.'s method [SBLM11] only keeps a relatively fixed pattern due to the limit of the employ clothoid curves. Similarly, because the dynamic turning angle is used to form the constraints in the lateral direction in both the existing methods, $v_{lateral}$ of the subject vehicles by them have similar patterns (i.e., first increasing and then decreasing), which is quite different from the measured ground-truth data (Fig. 6(c) and (f)). Also, it is noteworthy that, since Shen and Jin's method [SJ12] uses a car-following model to compute its velocity during lane-changing, its performance highly depends on the predefined parameter values of the used car-following model such as the desired vehicle speed, comfortable deceleration, and so on.

Fig. 7 further compares our method with the same two methods

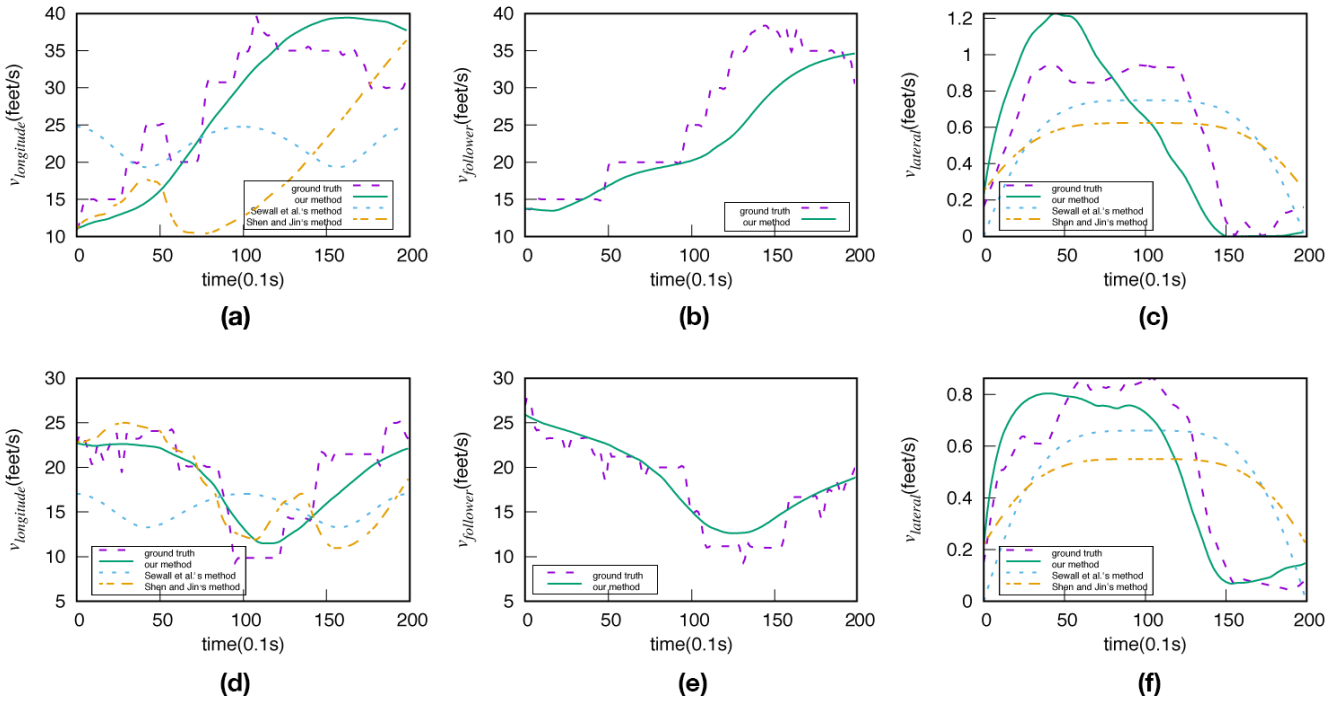


Figure 6: *Comparisons of the ground-truth, our method, and two existing methods [SBLM11, SJ12] for two selected examples: V2719 (top row) and V2823 (bottom row). The purple curves denote the ground-truth trajectory from the NGSIM dataset, the green curves denote the simulation results by our method, and the blue and yellow curves denote the results by [SBLM11, SJ12].*

in terms of additional statistics: positions in the longitude/lateral directions ((a) and (d)), the subject vehicle's gap to the leader vehicle during lane-changing ((b) and (e)), the follower vehicle's gap to the subject vehicle during lane-changing ((c) and (f)). We can see that the results by our method is clearly closer to the ground-truth than the other two methods. Also, when the subject vehicle is doing lane-changing, the follower vehicle always keeps a safe gap with it and the gap increases gradually (Fig. 7(c) and (f)). Note that both Sewall et al.'s method and Shen and Jin's method cannot calculate the velocity of the follower vehicle; instead, they use car-following model to calculate its velocity. Therefore, the velocities of the follower vehicle in these two methods are not plotted in Fig. 7(c) and (f).

We also compared the visualized vehicle trajectories during lane-changing by our method and the ground-truth. Fig. 8 shows an example of side-by-side comparison between several selected frames of a ground-truth traffic segment and those simulated by our approach. In this example, the lane-changing vehicle trajectory is very close to the ground-truth. We also visually compared the lane-changing vehicle trajectories by our method and the two existing methods [SBLM11, SJ12], and our method generated more sound trajectories than the other two for all the test examples. Comparisons of their animation results can be found in the supplemental demo video.

In sum, our method can clearly outperform the two existing methods [SBLM11, SJ12] in terms of lane-changing simulation, due to the following main reasons.

- In our approach, the subject vehicle has more selection options to perform lane-changing, since its selection is not limited to the adjacent gap on the target lane, but also the forward and backward gaps.
- The two existing methods are only accurate at a coarse level, since they heavily depend on empirically-tweaked parameter values. By contrast, our method can obtain a better simulation accuracy, because it learns the behavior patterns of lane-changing vehicles automatically from real-world data.

We also tested our approach on a number of new virtual road networks (Fig. 9 shows several of them). In these experiments, we used our approach for lane-changing, and used the IDMM model described in [SJ12] for car-following. For animation results, please refer to the supplemental demo video.

8. Conclusion and Discussion

We present a new data-driven method to simulate the process of lane-changing in traffic simulation. Our method can make the subject vehicle to take account of more gap options on the target lane

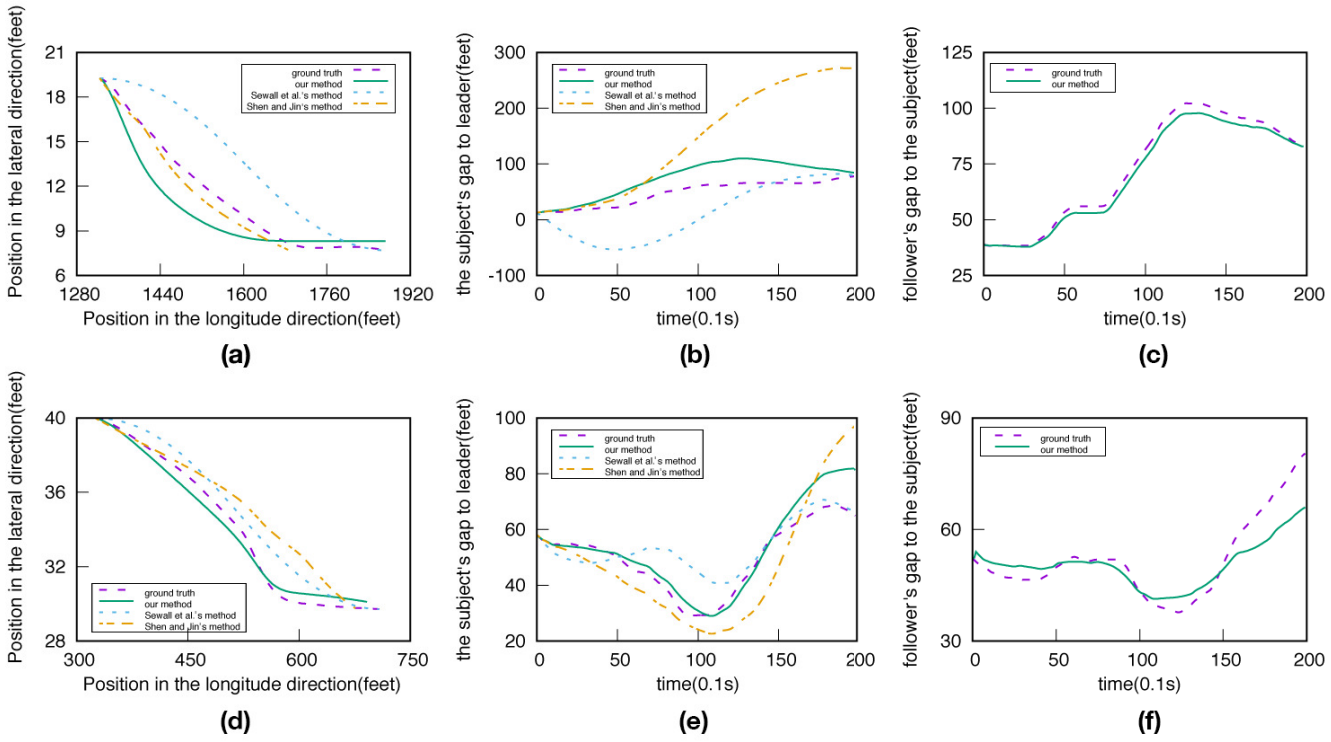
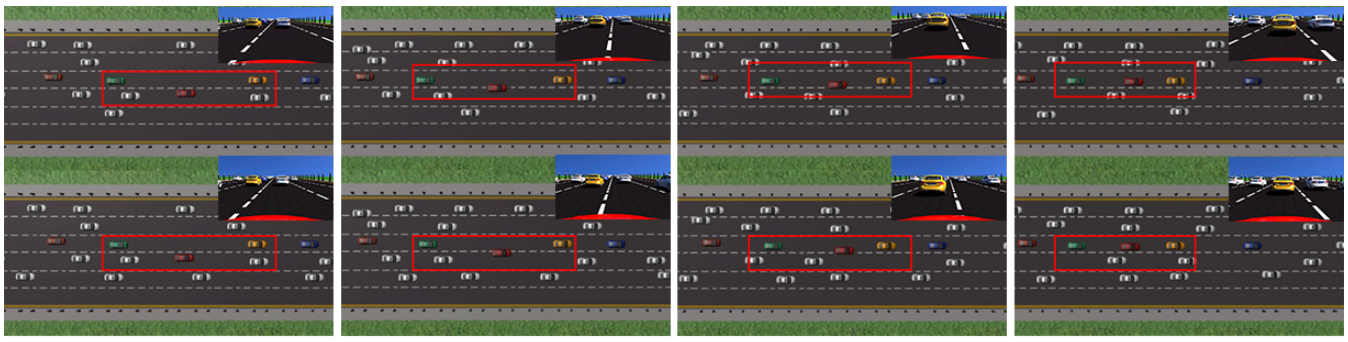


Figure 7: *Statistics comparisons of the ground-truth, our method, and two existing methods [SBLM11, SJ12] for two selected examples: V2719 (top row) and V2823 (bottom row). The purple curves denote the ground-truth trajectory from the NGSIM dataset, the green curves denote the simulation results by our method, and the blue and yellow curves denote the results by [SBLM11, SJ12].*



(a) 100th frame (b) 180th frame (c) 240th frame (d) 300th frame

Figure 8: *An example of side-by-side comparisons between several selected frames of a ground-truth traffic segment (top) and those simulated by our approach (bottom). The small top-right window in each panel shows the rendered traffic from the perspective of the driver of the lane-changing vehicle.*

to cut in as well as achieve more realistic lane-changing trajectories for the subject vehicle and the follower vehicle. Through many experiments and comparisons with two selected state-of-the-art methods, we show that our approach can soundly outperform them in terms of lane-changing simulation accuracy and quality. Our model can be flexibly used together with a variety of existing car-following models to produce natural traffic animations in various virtual environments.

Despite the demonstrated effectiveness of our method, it still has

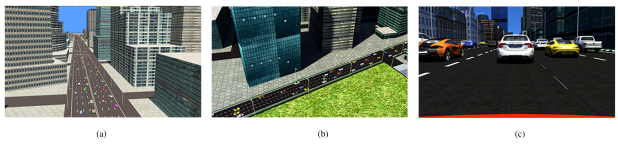


Figure 9: *Several virtual road environments used in our experiments*

a few limitations. Our current model is still not generalized enough to handle all kinds of lane-changing behaviors in traffic simulation. For example, if the velocities of the simulated traffic is close to those in the training dataset, our method can perform well. However, if they are quite different, the resulting lane-changing simulations by our method may not be satisfactory. Such a failure example is shown in Fig. 10, where the vehicle velocities are far beyond the velocity range in the training dataset, the predicted lane-changing behaviors are less accurate. On the other hand, we do not take driver variability into account.

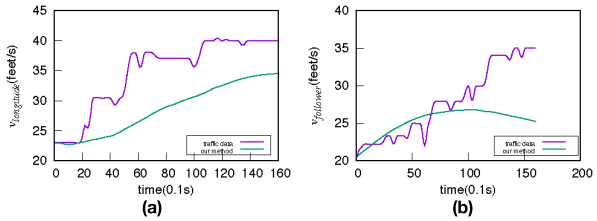


Figure 10: *Comparison between a simulated result by our method and the ground-truth data. (a): The velocity of the subject vehicle in the longitude direction, and (b): the velocity of the follower vehicle. This vehicle took 178 frames to complete the lane-changing. It is obvious that the predictions are less accurate after the 85-th frame.*

In this work, we mainly focus on simulating the lane-changing behavior of the subject vehicle and other surrounding vehicles on high ways. And our method is applicable to "regular traffic". "Irregular traffic" like countries where lanes might not be explicitly defined or roads with different surface are difficult to simulate. So we cannot directly apply it to simulate lane-changing on local roads such as local intersections. As our future work, we plan to extend this framework to learn individual vehicles' driving characteristics including the lateral and longitude motion in local roads, which would lead to more vivid traffic reconstruction and simulation for many applications.

References

[CAM11] CHONG L., ABBAS M. M., MEDINA A.: Simulation of Driver Behavior with Agent-Based Back-Propagation Neural Network. *Transportation Research Record: Journal of the Transportation Research Board 2249* (2011), 44–51. 2

[CAMH13] CHONG L., ABBAS M. M., MEDINA FLINTSCH A., HIGGS B.: A rule-based neural network approach to model driver naturalistic behavior in traffic. *Transportation Research Part C: Emerging Technologies 32* (2013), 207–223. 2

[CDJ15] CHAO Q., DENG Z., JIN X.: Vehicle–pedestrian interaction for mixed traffic simulation. *Computer Animation and Virtual Worlds 26*, 3-4 (2015), 405–412. 2

[Cri11] CRIMINISI A.: Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning. *Foundations and Trends in Computer Graphics and Vision 7*, 2-3 (2011), 81–227. 4

[CSJ13] CHAO Q., SHEN J., JIN X.: Video-based personalized traffic learning. *Graphical Models 75*, 6 (2013), 305–317. 2

[DBC09] DURET A., BUISSON C., CHIABAUT N.: Estimating Individual Speed-Spacing Relationship and Assessing Ability of Newell's Car-Following Model to Reproduce Trajectories. *Transportation Research Record: Journal of the Transportation Research Board 2088*, -1 (2009), 188–197. 3

[Ger55] GERLOUGH D. L.: *Simulation of Freeway Traffic on a General-purpose Discrete Variable Computer*. University of California, Los Angeles, 1955. 2

[HES14] HOU Y., EDARA P., SUN C.: Modeling mandatory lane changing using bayes classifier and decision trees. *IEEE Transactions on Intelligent Transportation Systems 15*, 2 (2014), 647–655. 2

[Hid05] HIDAS P.:. *Transportation Research Part C: Emerging Technologies 13*, 1 (2005), 37–62. 2, 3, 7

[HN89] HECHT-NIELSEN R.: Theory of the Backpropagation Neural Network. *Proceedings Of The International Joint Conference On Neural Networks 1* (1989), 593–605. 6

[KPAB15] KAPADIA M., PELECHANO N., ALLBECK J., BADLER N.: Virtual crowds: Steps toward behavioral realism. *Synthesis Lectures on Visual Computing 7*, 4 (2015), 1–270. 2

[KT08] KESTING A., TREIBER M.: Calibrating car-following models by using trajectory data: Methodological study. *Transportation Research Record: Journal of the Transportation Research Board 2088* (2008), 148–156. 3

[KTH07] KESTING A., TREIBER M., HELBING D.: General lane-changing model mobil for car-following model. *Transportation Research Record 1999*, 1 (2007), 86–94. 2

[LCX*14] LU X., CHEN W., XU M., WANG Z., DENG Z., YE Y.: AA-FVDM: An accident-avoidance full velocity difference model for animating realistic street-level traffic in rural scenes. *Computer Animation and Virtual Worlds 25*, 1 (2014), 83–97. 2

[LFCCO09] LERNER A., FITUSI E., CHRYSANTHOU Y., COHEN-OR D.: Fitting behaviors to pedestrian simulations. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2009), SCA '09, ACM, pp. 199–208. 2

[LW55] LIGHTHILL M. J., WHITHAM G. B.: On Kinematic Waves. II. A Theory of Traffic Flow on Long Crowded Roads. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 229*, 1178 (1955), 317–345. 2

[MW12] MENG Q., WENG J.: Classification and regression tree approach for predicting drivers' merging behavior in short-term work zone merging areas. *Journal of Transportation Engineering 138*, 8 (2012), 1062–1070. 2

[MWDW15] MAO T., WANG H., DENG Z., WANG Z.: An efficient lane model for complex traffic simulation. *Computer Animation and Virtual Worlds 26*, 3-4 (2015), 397–403. 2

[MWLT13] MA C., WEI L.-Y., LEFEBVRE S., TONG X.: Dynamic element textures. *ACM Trans. Graph. 32*, 4 (July 2013), 90:1–90:10. 2

[New61] NEWELL G. F.: Nonlinear Effects in the Dynamics of Car Following. *Operations Research 9*, 2 (1961), 209–229. 2

[ngs] Ngsim. http://ngsim-community.org/. 2013. 3

[NS92] NAGEL K., SCHRECKENBERG M.: A cellular automaton model for freeway traffic. *Journal of Physics I 2*, 2 (1992), 2221–2229. 2

[Pay71] PAYNE H. J.: *Models of Freeway Traffic and Control*. Simulation Councils, Incorporated, 1971. 2

[PBC09] PUNZO V., BORZACCHIELLO M. T., CIUFFO B. F.: Estimation of vehicle trajectories from observed discrete positions and next-generation simulation program (ngsim) data. In *Proc. of Transportation Research Board 88th Annual Meeting* (2009), p. 17. 3

[PF60] PRIGOGINE I., F.C.ANDREWS: A Boltzmann-Like Approach for Traffic Flow. *Operations Research 8*, 6 (1960), 789–797. 2

[SBLM11] SEWALL J., BERG J. V. D., LIN M. C., MANOCHA D.: Virtualized traffic: reconstructing traffic flows from discrete spatiotemporal data. *IEEE transactions on visualization and computer graphics 17*, 1 (2011), 26–37. 1, 2, 7, 8, 9

[SH99] SHVETSOV V., HELBING D.: Macroscopic dynamics of multilane traffic. *Physical review E 59*, 6 (1999), 6328. 2

[SJ12] SHEN J., JIN X.: Detailed traffic animation for urban road networks. *Graphical Models 74*, 5 (2012), 265–282. 2, 7, 8, 9

[SWL11] SEWALL J., WILKIE D., LIN M. C.: Interactive hybrid simulation of large-scale traffic. *Proceedings of the 2011 SIGGRAPH Asia Conference on - SA '11 30*, 6 (2011), 1. 1

[SWML10] SEWALL J., WILKIE D., MERRELL P., LIN M. C.: Continuum traffic simulation. *Computer Graphics Forum 29*, 2 (2010), 439–448. 2

[TH02] TREIBER M., HELBING D.: Microsimulations of freeway traffic including control measures. *Automatisierungstechnik*, 49 (2002), 478–484. 2

[THH00] TREIBER M., HENNECKE A., HELBING D.: Congested traffic states in empirical observations and microscopic simulations. *Physical Review E 62*, 2 (2000), 1805. 2

[Whi11] WHITHAM G. B.: *Linear and Nonlinear Waves*. Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts. Wiley, 2011. 2

[WSL13] WILKIE D., SEWALL J., LIN M.: Flow reconstruction for data-driven traffic animation. *ACM Transactions on Graphics 32* (2013), 1. 1