PATROL: A Velocity Control Framework for Autonomous Vehicle via Spatial-Temporal Reinforcement Learning

Zhi Xu[1#], Shuncheng Liu[1#], Ziniu Wu[2,3], Xu Chen[1], Kai Zeng[3], Kai Zheng[1*], Han Su[1,4*]
[1]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China
[2]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Boston, USA
[3]Alibaba Group, Hangzhou, China
[4]Yangtze Delta Region Institute, University of Electronic Science and Technology of China, Quzhou, China
zhixu023@std.uestc.edu.cn,liushuncheng@std.uestc.edu.cn,ziniuw@mit.edu,xuchen@std.uestc.edu.cn
zengkai.zk@alibaba-inc.com,zhengkai@uestc.edu.cn,hansu@uestc.edu.cn

ABSTRACT

The largest portion of urban congestion is caused by 'phantom' traffic jams, causing significant delay travel time, fuel waste, and air pollution. It frequently occurs in high-density traffics without any obvious signs of accidents or roadworks. The root cause of 'phantom' traffic jams in one-lane traffics is the sudden change in velocity of some vehicles (i.e. harsh driving behavior (HDB)), which may generate a chain reaction with accumulated impact throughout the vehicles along the lane. This paper makes the first attempt to address this notorious problem in a one-lane traffic environment through velocity control of autonomous vehicles. Specifically, we propose a velocity control framework, called *PATROL* (sPAtial-Temporal ReinfOrcement Learning). First, we design a spatial-temporal graph inside the reinforcement learning model to process and extract the information (e.g. velocity and distance difference) of multiple vehicles ahead across several historical time steps in the interactive environment. Then, we propose an attention mechanism to characterize the vehicle interactions and an LSTM structure to understand the vehicles' driving patterns through time. At last, we modify the reward function used in previous velocity control works to enable the autonomous driving agent to predict the HDB of preceding vehicles and smoothly adjust its velocity, which could alleviate the chain reaction caused by HDB. We conduct extensive experiments to demonstrate the effectiveness and superiority of *PATROL* in alleviating the 'phantom' traffic jam in simulation environments. Further, on the real-world velocity control dataset, our method significantly outperforms the existing methods in terms of driving safety, comfortability, and efficiency.

CCS CONCEPTS

• **Applied computing** → **Transportation**; • **Information systems** → *Spatial-temporal systems*; • **Theory of computation** → *Sequential decision making*.

KEYWORDS

velocity control, reinforcement learning, autonomous vehicle

ACM Reference Format:
Zhi Xu[1#], Shuncheng Liu[1#], Ziniu Wu[2,3], Xu Chen[1], Kai Zeng[3], Kai Zheng[1*], Han Su[1,4*]. 2021. PATROL: A Velocity Control Framework for Autonomous Vehicle via Spatial-Temporal Reinforcement Learning. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482283

1 INTRODUCTION

With the rapid growth in urbanization, most major cities around the world suffer from traffic congestion, resulting in serious travelling inefficiency [13], significant fuel waste and air pollution [11]. It is reported that in the US alone, urban congestion costs 8.8 billion hours of travel delay, and 3.3 billion gallons of wasted fuel per year [27], the largest portion of which is caused by the **'phantom' traffic jam** phenomenon [8, 17, 22]. It is a type of traffic jams commonly encountered in high-density traffics, without any obvious signs such as accidents, roadwork, closed lanes and so on. The root cause of 'phantom' traffic jam is the sudden changes in velocity (such as sudden braking and accelerating) of some vehicles [9], which are referred to as *harsh driving behavior* (HDB) in the field of transportation. In high-density traffic, one driver's HDB may generate a 'butterfly effect', create a chain reaction for the following vehicles, causing a series of sudden brakes and accelerations. This chain reaction will propagate with increasing delay time for the vehicles along the lane, leading to traffic congestion. Solving the phantom traffic jams is of vital importance to improving the urban traffic condition and fuel efficiency.

Traditionally, dealing with the 'phantom' traffic jams is almost impossible because it is impossible to regulate human drivers' behaviors. The rapid development of autonomous vehicles and advanced assisted driving technology provides a promising way to solve the 'phantom' traffic jam challenges by controlling the velocity of the one autonomous vehicle in the lane and breaking the

[#]The first two authors contribute equally to this paper.
[*]Corresponding authors: Kai Zheng and Han Su.

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482283

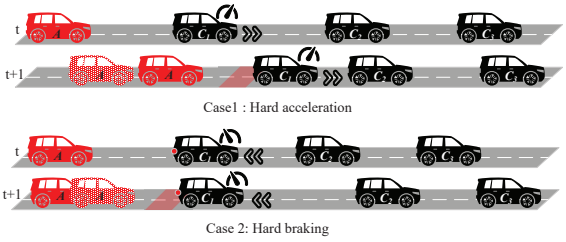


Figure 1: Two common harsh driving behaviors in a single lane. The red car is controlled by velocity control models, the black cars is the conventional cars, the red area of the road represents unsafe following distance. The results of time $t+1$ describe the difference between our model and the existing models, in which the red car with dotted lines is controlled by our model.

entire chain reaction. Unfortunately, the existing velocity decision (i.e. velocity control) algorithm of autonomous vehicles can not effectively address this problem. Specifically, the velocity decision models can be classified into two categories: *imitation-based*, and *car-following based*. The imitation based models, such as [36], take a supervised learning approach. They aim at learning ML models to minimize the difference between the behavior of autonomous vehicles and the labeled human driver behaviors, trying to make autonomous vehicles mimic the travel patterns of human-driven vehicles. Therefore, the imitation based approaches heavily rely on real human driving behaviors as training data, which inevitably imitate the human HDBs in the training data. On the other hand, the car-following based models try to follow the vehicle in front by maintaining a safe distance, and at the same time optimize the efficiency and comfortability of the autonomous vehicle using either rule-based or ML-based methods. Since these approaches only consider the state of the car immediately in front, ignoring the cars further ahead and their historical travel behaviors in, vehicles controlled by these algorithms will imitate the HDBs of the car ahead of it.

We provide an example in Figure 1 to illustrate the limitation of existing velocity decision algorithms. In Case 1, car C_1 accelerates continuously without maintaining a proper distance from the cars in front. Obviously, car A should not accelerate to follow car C_1 because C_1 will have to brake in the near future to avoid a collision. However, since the existing velocity control methods can only observe the state of the first car immediately ahead, they will instruct car A to blindly follow car C_1 to accelerate. In Case 2, car B ahead suddenly brakes despite that the overall traffic ahead is relatively smooth. Under the existing velocity control methods, car A will take the same sudden brake. Yet, a more reasonable way is to gradually slow down and wait for the stabilization of the traffic flow ahead. In both cases, a chain reaction would occur if every car in the lane blindly follows the car ahead. Specifically, if a car traveling at a constant velocity suddenly brakes and it takes x seconds to recover back to its previous velocity, it will take more than x seconds for the car behind it to recover since there naturally exists a delay in human reaction. This effect will propagate and accumulate along with the cars in the lane, causing the 'phantom' traffic jams.

This paper makes the first effort to solve the notorious 'phantom' traffic jam problem through the velocity control of autonomous vehicles. Specifically, we focus on a simplified environment of a single lane with no overtaking. In our PATROL framework, we propose a novel reinforcement-learning-based velocity control model, called *LADDPG* (Deep Deterministic Policy Gradient equipped with LSTM and Attention mechanisms) to tackle the 'phantom' traffic jam problem. In order to alleviate the chain reaction caused by HDB, we find that the autonomous driving vehicle must gather information of various vehicles ahead across various historical time steps to detect the abnormal behavior and adjust its velocity accordingly and smoothly. Therefore, our LADDPG method first formulates this information into a spatio-temporal graph and then designs a novel model architecture to process this graph. To distill useful spatial information from the graph, we propose two attention mechanisms VR-Att and DR-Att to learn the interactions and effects of different vehicles ahead on the target autonomous vehicle. To extract meaningful temporal information from the graph, we propose a LSTM structure to understand the historical driving behaviors of these vehicles and detect their abnormal behaviors (HDB events) beforehand. These novel model designs in LADDPG bring the autonomous vehicle a wholistic view of the driving patterns of multiple vehicles ahead along the lane and equip it with the ability to detect and predict HDB events of these vehicles in order to adjust its driving velocity smoothly. We conduct extensive experiments to demonstrate the superiority of our method in alleviating the 'phantom' traffic jam in simulation environments. Further, on real-world velocity control datasets, our method significantly outperforms the existing methods in terms of driving safety, comfortability and efficiency.

In summary, our contributions are listed as follows.

- We identify the importance of the 'phantom' traffic jam problem and make the first effort to tackle this problem through velocity control.
- We propose a novel velocity control model LADDPG for autonomous vehicles to break the chain reaction caused by HDB, thus alleviating the 'phantom' traffic jam.
- We conduct extensive experiments to evaluate the effectiveness of our method.

2 BACKGROUND AND RELATED WORK

The largest portion of urban congestion is caused by 'phantom' traffic jams, which frequently occur in high-density traffics. In a specific one-lane dense-traffic scenario, a 'phantom' jam begins when a vehicle hits the brake too hard or gets too close to the leading vehicle (HDB event), which causes the chain reaction that the vehicle follows this vehicle slows down even more. This slowdown in speed creates a chain reaction that propagates backward throughout the lane with increasing delay, causing the 'phantom' traffic jam. It has a notorious effect on traveling time and fuel waste, so it is particularly meaningful to alleviate or solve the 'phantom' traffic jam problem. In this paper, we consider optimizing the velocity of autonomous vehicles to deal with the chain reaction caused by HDB in a one-lane road, so as to alleviate 'phantom' traffic jams and improve road capacity.

In the field of autonomous driving, numerous researches have proposed to study the impact of controlling autonomous vehicles' velocity on the stability of traffic flow [7, 30, 37]. Specifically, the

velocity control problem describes the response in driving speed of an autonomous vehicle with respect to the traffic conditions ahead. In the following, we provide the details of the three kinds of existing velocity control methods.

Traditional Methods: The traditional velocity control models consider the information (i.e. distance and velocity difference) between two consecutive vehicles, then use specific mathematical functions to map the information into an optimal velocity decision. Specifically, [6, 29] deign linear dynamical equations to model Adaptive Cruise Control (ACC) i.e. the autonomous vehicle keeping the constant distance with the preceding vehicle. [4, 32, 33] propose intelligent driver models (IDM), which considers the expected velocity, the expected following distance, and additional restrictions (i.e. the minimum spacing, maximum vehicle acceleration). Unfortunately, the traditional velocity control models have inherent shortcomings: mathematical equations or rules can hardly consider all influencing factors and generalizable to rare cases, so it is difficult to apply to real-world complex scenarios.

Machine Learning (ML) based Methods: With the rapid development of ML techniques, many researches adopt data-driven ML methods to imitate human driver's behavior from the massive real-world field data. For example, [36, 42] uses the deep neural networks to model the historical trajectory data and predict the acceleration of the vehicle in the next time step. [18, 41] present a learning-based framework for autonomous driving which can learn from human demonstrations, replicating what a human driver would do in the same situation to output acceleration through estimating the relevant features representing the driving situation. Although these ML-based models have produced great achievements in microscopic traffic simulation, they have two key limitations when applied to autonomous vehicle planning: 1) the learning-based network structures will inevasible modeling errors, which can lead to dangerous traffic accidents; 2) the training label of human driving data may have inappropriate behaviors (i.e. HDB), which affects the stability of traffic flow.

Reinforcement Learning (RL) based Methods: In order to address the above existing limitations of ML-based methods, many researches use deep RL algorithms to deal with the velocity control problem. The key idea of these methods is to enable the free exploration of the RL agent to form a driving pattern that targets at optimizing certain goals (e.g. driving safety, comfortability, and efficiency). For example, [12, 40] adopts a reinforcement model to output the acceleration (or deceleration) that maximizes vehicle velocity along the path, without losing its dynamic stability. [39] uses the double deep-Q-network to control the velocity of the autonomous vehicle, using the information of a lead and following car and historical driving data as model' input to achieve a human-like learning effect. [43, 44] applies the deep deterministic policy gradient (DDPG) and designs a novel reward function to model the velocity of autonomous vehicles. It is currently the state-of-the-art algorithm that the vehicle can drive safely, efficiently, and comfortably. The reward function of DDPG includes three indicators: Time-to-Collision (TTC), Time headway (THW), and Jerk. In detail, TTC is a safety indicator and represents the time that remains until a collision between two vehicles would have occurred if the collision course and velocity difference are maintained [2]. THW is an important indicator that reflects the road capacity and can be represented by the time-value derived from measures of the following vehicle's velocity and spacing [38]. Jerk is a physical quantity related to acceleration, which describes how fast the acceleration of a vehicle changes and can be obtained by calculating the derivative of acceleration [10], can be used to measure driving comfort [14].

Table 1: Summary of Notations

Notation	Definition
C_i	A conventional vehicle C_i
A_i	An autonomous vehicle A_i
I	Total number of vehicles on the road
T	The time duration of interest
t	A time step $t \in T$
$V_{A_i}^t$	The velocity of the vehicle A_i at time t
$D_{A_i}^t$	The longitudinal distance of vehicle A_i at time t
$D_r^t(C_i, C_j)$	The relative distance between C_i and C_j at time t
$V_r^t(C_i, C_j)$	The relative velocity between C_i and C_j at time t
λ	The length of historical time steps
m	The number of vehicles ahead observed by the autonomous vehicle

Unfortunately, the aforementioned RL methods have one common problem that they only consider the first car ahead at one time step, which makes the RL agent unable to detect the abnormal behavior of traffic ahead of time and blindly follows the previous car. Thus, they can not solve the chain reaction caused by HDB, which may lead to the 'phantom' traffic jams.

Our Solution: We propose a novel velocity control framework to solve 'phantom' traffic jams. We consider an autonomous vehicle controlled by our model in an interactive one-lane traffic environment. With the development of perception technology, the autonomous vehicle A can obtain the information(velocity and real-time location) of multiple vehicles ahead through its sensors or IoV (Internet of Vehicles).

We slightly modify the reward function (TTC+THW+Jerk) of DDPG so that our method can effectively alleviate the chain reaction caused by HDB. The Jerk reward can penalize large acceleration or deceleration (HBA) but it is suppressed by the other two rewards in the original DDPG algorithm [3, 43, 44]. Specifically, if the vehicle in front of the agent autonomous vehicle suddenly brakes, the agent must brake immediately to maintain a safe distance (i.e. TTC suppresses Jerk). If the vehicle in front suddenly accelerates, the agent will accelerate immediately to ensure driving efficiency (i.e. THW suppresses Jerk). In our LADDPG, we increase the relative weight of Jerk to eliminate HBA behaviors. When considering vehicles ahead for multiple time steps, the large Jerk reward will enforce the agent to understand the driving patterns of vehicles in front, predict their behaviors, and decide its own velocity with smooth changes to avoid HBA. Specifically, if the agent observes a sudden brake of a vehicle in the lane (e.g. two vehicles in front of it), it should expect a sudden brake of the vehicle immediately in front of it. In this case, the agent has to decelerate smoothly before this vehicle hits the brake and all three reward will be high for this behavior. After the RL agent has explored enough situations like

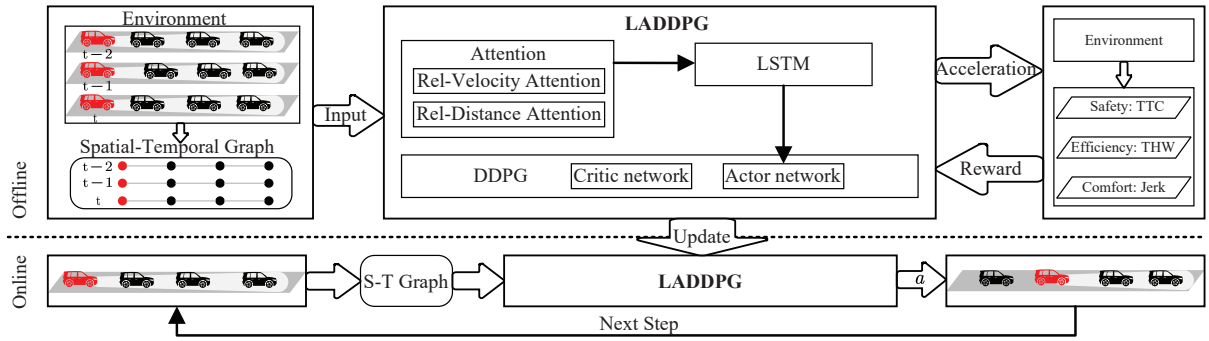


Figure 2: PATROL overview

this, the reward function will guild this autonomous agent to fully understand the traffic dynamic and thus avoid HBA and alleviates the 'phantom' traffic jam.

3 METHOD

Existing works on velocity decisions only consider the action of one vehicle ahead in one time step [39, 44], ignoring the vehicles further ahead and their historical travel behaviors. If a vehicle in front of the autonomous vehicle breaks or accelerates rapidly (HDB events), these velocity decision methods will likely mimic this behavior leading to a serious chain reaction to the vehicles behind it, namely 'phantom' traffic jams.

In this section, we propose an improved reinforcement learning model LADDPG to output the optimal velocity decision of the autonomous vehicle based on its surrounding traffic circumstance at each time step. Our newly proposed method not only ensures the smooth, safe and efficient driving pattern but also alleviates the chain reaction caused by HDB events.

3.1 Framework Overview

As shown in Figure 2, our framework is composed of an offline training phase and an online inference phase. In the offline phase, we first extract informative features (e.g. velocity and distance of the vehicles ahead) and structure them into a spatial-temporal graph (detailed in Section 3.2). Then, we use the LSTM-and-attention-based actor network to learn the spatial-temporal interaction from this graph and output the optimal velocity decision accordingly (detailed in Section 3.3). At last, we use the predefined reward function to evaluate the safety, efficiency, and stability of the outputted decision and use the feedback signals to train the model. In the online phase, the agent will observe its surrounding traffic environment and use the trained model to make decisions in real time.

3.2 Reinforcement Learning Agent Design

First, we introduce the state, action, state updating, and reward function of our LADDPG model.

State: The state represents the environmental information obtained by the agent, which in our method consists of the relative velocity and distance of multiple vehicles ahead of it. In order to alleviate the chain reaction of HDB events, the agent needs to consider the information of multiple vehicles ahead at multiple historical time steps to smoothly accelerate or decelerate so that its driving velocity will not cause further HDB of the vehicles behind it. In order to capture the spatial and temporal dynamic changes of the vehicles in front of the autonomous agent, we use a **spatial-temporal graph** to represent the state at each time step.

A spatial-temporal graph is composed of spatial edges along with historical time steps. At time step t, the state s_t is a spatial-temporal graph, containing λ historical time steps spatial features, i.e. $[t, t-1, ...t-\lambda+1]$. The autonomous vehicle A can detect m vehicles ahead, i.e. $\{C_i\}$ where $i \in [1, m]$. Naturally, they formed a fleet, where C_0 (the agent A itself) is the last vehicle and C_m is the first vehicle. At each historical time step $\tau \in [t, t-1, ...t-\lambda+1]$, we set each vehicle C_i^τ as the a node in the graph and represent its feature as a concatenation of its current velocity $V_{C_i}^\tau$ and longitudinal distance $D_{C_j}^\tau$ w.r.t. the agent. Furthermore, we create a spatial edge $e_{i,i+1}^\tau$ between two adjacent vehicles C_i and C_{i+1}, which contains the interactive information between these two vehicles at each historical time τ. Specifically, $e_{i,i+1}^\tau$ can be represented as the relative velocity and distance between the two adjacent vehicles:

$$e_{j,j+1}^\tau = [V_r^\tau(C_j, C_{j+1}), D_r^\tau(C_j, C_{j+1})], \tag{1}$$

where $V_r^\tau(C_j, C_{j+1}^\tau) = V_{C_j}^\tau - V_{C_{j+1}}^\tau$, and $D_r^\tau(C_j, C_{j+1}^\tau) = D_{C_j}^\tau - D_{C_{j+1}}^\tau$.

Therefore, at time step t, the state s_t is represented as:

$$s_t = \begin{bmatrix} e_{0,1}^{t-\lambda+1} & e_{1,2}^{t-\lambda+1} & \cdots & e_{m-1,m}^{t-\lambda+1} \\ \vdots & \vdots & \ddots & \vdots \\ e_{0,1}^{t} & e_{1,2}^{t} & \cdots & e_{m-1,m}^{t} \end{bmatrix}_{\lambda \times 2m} \tag{2}$$

The spatial-temporal graph is scalable, according to the number of vehicles ahead (m) and the length of historical time steps (λ).

Action: Given the state s_t, the agent will decide its optimal driving velocity decision at time step t, denoted as action a_t. Following prior works [44], the a_t represents the longitudinal acceleration of the agent, and it is bounded between -3 to $3m/s^2$.

State Updating: Given a current state s_t and a chosen action a_t, the velocity and the longitudinal distance of the autonomous vehicle in the next time step can be calculated as follows:

$$\begin{aligned} V_{C_0}^{t+1} &= V_{C_0}^t + a_t * \Delta t \\ D_{C_0}^{t+1} &= D_{C_0}^t + V_{C_0}^t * \Delta t + 0.5 * a_t * (\Delta t)^2 \end{aligned} \tag{3}$$

where Δt is the time interval between two consecutive time steps, which is 0.1 second in our case. In addition to the autonomous vehicles controlled by our model, the features (velocity and distance) of other vehicles can be obtained directly from the environment. Thereafter, the state s_{t+1} can be efficiently updated.

Reward: The reward value is associated with the each transition from s_t to s_{t+1} after taking action a_t to evaluate the quality of this action. We adopt a recently proposed reward function [44] to evaluate the agent travel safety (TTC), comfortability (Jerk) and efficiency (THW). Since this reward function is not sensitive to the harsh acceleration and deceleration of the agent, we define a weighted version of this reward function by setting the weights as tunable hyperparameters. The reward value at time step t is defined as follows:

$$r_t = W_1 * F(x_t) + W_2 * G(y_t) + W_3 * H(z_t) \tag{4}$$

where the W_1, W_2, W_3 are hyper-parameters, which are discussed in the experiment. The x_t, y_t, z_t represent the value of $Jerk$, TTC and THW at time step t respectively, and the F, G, H functions are defined as follows:

$$F(x_t) = -(\frac{x_t}{B})^2 \tag{5}$$

$$G(y_t) = \begin{cases} \log(\frac{y_t}{4}), & 0 \le y_t \le 4, \\ 0, & otherwise. \end{cases} \tag{6}$$

$$H(z_t) = \frac{1}{z_t \sigma \sqrt{2\pi}} \exp \frac{-(\ln z_t - \mu)^2}{2\sigma^2} \tag{7}$$

where B is used to normalize the value into the range of $[0, 1]$, and can be computed by empirical data. According to the data, the μ and σ equals to 0.3052 and 0.2031 respectively. Specially, if there is a collision, the episode will terminate and get a large penalty (i.e, $r_t = -100$).

3.3 Model Details

Our neural architecture is based on a deep determined policy gradient (DDPG) approach. It is comprised of two networks: the actor network and the critic network (see Figure 3). The details are as follows:

Actor Network: The actor network maps the current state s_t of the environment to the optimal action and guides the agent's driving velocity. The architecture of actor networks in previous works is relatively simple because they only consider the first vehicle ahead for one time step as their input state. However, our state inputs (i.e. the spatial-temporal graph described in the previous section) need more advanced techniques to process the underlying interactions within the graph and extract meaningful information from it. To accomplish this goal, we propose two important improvements over the existing actor networks.

The first improvement is designed to tackle spatial features, i.e. understanding the interactions of various vehicles ahead of the agent, which has a different impact on the agent. As shown in the examples of Figure 1, only when the velocity decision is made by considering the influence of multiple vehicles in front of the autonomous vehicle A simultaneously, can the chain-reaction be effectively alleviated. Therefore, we use global **attention mechanism** to process the spatial-temporal graph, which allocates different weights to the edges to capture the influence of different vehicles

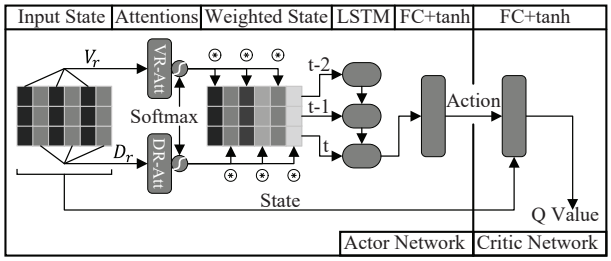


Figure 3: LADDPG overview

ahead on the agent and learn their interactions adaptively. The structure of the attention mechanism is shown in Figure 3. Since the relative velocity and relative distance are the two independent features, we design two attention blocks, namely relative velocity attention (VR-Att) and relative distance attention (DR-Att), to handle them separately. The two blocks have the same network structure, so we will only elaborate VR-Att in detail. Specifically, for each historical time step τ, we first feed the relative velocities of all vehicles ahead into the two fully connected layers to calculate the weight of relative velocity on each edge of the spatial-temporal graph, and then we multiply the weight with the relative velocity value to get the weighted feature. The calculation process is formulated as follows:

$$h_v^\tau = \tanh\left(W_1 \cdot [V_r^\tau(A, C_1), \ldots, V_r^\tau(C_{m-1}, C_m)]\right) \tag{8}$$

$$weight_v^\tau = softmax(W_2 \cdot h_v^\tau) \tag{9}$$

$$\begin{aligned} &\widetilde{V_r}^\tau(A, C_1), \ldots, \widetilde{V_r}^\tau(C_{m-1}, C_m) = \\ &weight_v^\tau \cdot [V_r^\tau(A, C_1), \ldots, V_r^\tau(C_{m-1}, C_m)] \end{aligned} \tag{10}$$

where W_i denotes parameters of the network, and the tanh and $softmax$ are the activation function of the network layer. We can calculate the weight of relative distance $weight_d^\tau$ using the same procedure. Thereafter, we multiply the original relative velocity and distance features with the corresponding weights. At last, we feed all weighted edges for each historical time step $\tau \in [t, t-1, \ldots t-\lambda+1]$ as the weighted state to the next layer.

The second improvement is to deal with temporal features, i.e. capturing the historical driving patterns of these vehicles ahead of the agent. The dynamic change of traffic conditions ahead over time is very important for the agent to discover the abnormal behaviors (e.g. HDB) beforehand and make a reasonable velocity decision. Therefore, we leverage the Long short-term memory (LSTM) model to extract the temporal dynamics from weighted state for $\tau \in [t, t-1, \ldots t-\lambda+1]$. The output of the LSTM will be fed into a fully connected layer with tanh activation function as the optimal action a_t of the agent.

Critic Network: The critic network computes the value of an action at a state. It receives the current state s_t of the environment and the action a_t generated by the actor and outputs the Q-value associated with them. Following the design of the critic network in previous studies [21], the critic network contains three layers: an input layer, a hidden layer, and an output layer. The calculation can be expressed as the following equation 11.

$$Q = \tanh(W_0, relu(W_1, s_t, a_t)) \tag{11}$$

Where the W represents the different parameters. tanh and relu are the activate function.

3.4 Offline Training

The core problem of a RL task is to find an optimal policy for the agent, which corresponds to a function π that specifies the action that the agent should choose when at a specific state so as to maximize the accumulative rewards. In this paper, we adopt DDPG as the function approximators to solve the problem. The actor function $\mu(s|\theta^{\mu})$ which specifies the current policy by deterministically mapping states to a specific action. The critic function $Q(s, a|\theta^{Q})$ is learned using the Bellman equation as in Q-learning [23]. In addition, two target networks $\mu'(s|\theta^{\mu'})$ and $Q'(s, a|\theta^{Q'})$ are created for the main actor and critic networks respectively, to avoid divergence of the algorithm. In the process of training, we store the agent's experiences $e_t = (s_t, a_t, r_t, s_{t+1})$ in a replay buffer $D_t = \{e_1, \ldots, e_t\}$. At each time step, we randomly sample a minibatch of N transitions $\{(s_i, a_i, r_i, s_{i+1})\}_{i=1}^{N}$. The $Q(s, a|\theta^{Q})$ update at iteration i uses the following loss function $L(\theta^{Q})$:

$$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'}) \tag{12}$$

$$L(\theta^{Q}) = \frac{1}{N}\sum_{i}(y_i - Q(s_i, a_i|\theta^{Q}))^2 \tag{13}$$

In which the y_i of equation 12 is the the Bellman equation, the γ is the discount factor determining the agent's horizon [20]. Then, the parameters θ^{μ} of actor network are modified in a direction that is more likely to get a large Q-value. For the target networks. The weights of them are updated by having them slowly track the learned networks: $\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}$, $\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'}$ with $\tau = 0.001$. This means that the target values are constrained to change slowly, significantly improve the stability of learning.

4 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate our framework on simulated and real data, comparing against the performance of different methods from multiple metrics. we first design a simulator to study the ability of the controlled vehicle to deal with the chain reaction caused by harsh driving behaviors, which is the root cause of 'phantom' traffic jam in a one-lane road. Then, we study the safety, efficiency, and comfortability of the LADDPG model on the real-world data, demonstrating the effectiveness of the components in the LADDPG.

4.1 Experimental Settings

Simulator Setting: Cellular automata is widely used in traffic flow simulation [5, 24–26]. However, the traditional cellular automata construct a discrete space, leading to a large error in the speed control task. In this study, we define a continuous version of the cellular automata to simulate the traffic flow in a single lane. In the simulator, the time interval between two consecutive time steps is 0.1s. At each time step t, the status of the i$-th$ vehicle contains the following information: (current vehicle ID, velocity, longitudinal distance, ID of the vehicle in front of the current vehicle). In addition, the simulator uses periodic boundary, i.e., the number of vehicles is fixed and no vehicles will drive into or drive off the road. We set the traffic restrictions as $V_{max} = 80km/h$, $V_{min} = 0km/h$, and the other restrictions will be set in the simulation experiment part, such as the length of the road (km), traffic density (vehicles/km) and the density of the harsh driving behaviors (HDBs/km). In particular, autonomous vehicles are controlled by the velocity control models, non-autonomous vehicles are controlled by the HDM (Human Driver Model), which mimics the human-driver characteristics [34].

Table 2: The hyper-parameters setting

Parameter	Value
Learning rate for actor networks	0.001
Learning rate for critic networks	0.001
Reward discount factor γ	0.9
Batch size	1024
The size of replay buffer	20000
The coefficients of reward function W_1, W_2, W_3	1.2, 0.9, 9.4
The length of historical time interval λ	3
The number of Vehicles ahead the agent observes m	3

Real-World Dataset: We train and test our model on the next generation simulation (NGSIM) trajectory data [16], which is widely used in the study of traffic flow theory [31]. The NGSIM dataset provides longitudinal positional and velocity information for all vehicles in certain regions and contains variously complex traffic scenes, such as uncongested and congested traffic environments. To facilitate the training process, we preprocess the dataset into events, which can be formed as follows: 1) In an event, there are $m+1$ vehicles drive in the same lane, and there is no lane change behavior. 2) The duration of each event must be more than 15 seconds. Finally, we totally extracted 1200 valid events and divided them into the training set (840 events) and the test set (180 events) , and the validation set (180 events).

Implementation Details: Our proposed model LADDPG has two networks. One is the actor network, which is composed of attention mechanism, LSTM, and fully connected layer, and the number of neurons in the hidden layer is set as 32, 64, and 1 respectively. Another is the critical network, which consists of two fully connected layers, and the number of neurons in the hidden layer is set as 64 and 1 respectively.

We set the hyper-parameters of the model in table 2. The first five parameters are selected by the grid search method, and we set the best one as the default values. The hyper-parameters of the spatial-temporal graph (i.e., λ and m) are studied in Section 4.3. Further, LADDPG is trained on real-world train data set, tested on the simulator and the real-world test data set, the hyper-parameters of the model are adjusted on real-world validation data set. Finally, Our network is implemented in Python and runs on an Intel (R) CPU i7-4770@3.4GHz and 32G RAM.

Compared Methods: We compare our method with the following methods:

- **MPC-ACC** [29]. Adaptive cruise control (ACC) is an advanced driver-assistance system for road vehicles that can automatically control the velocity to maintain a safe distance from the vehicles ahead. In addition, the Model Predictive Control (MPC) technique is equiped for improving the controlling performance.
- **DDPG** [44]. The authors proposed a reinforcement-learning-based velocity control model (DDPG) and designed an effective

reward function, which can ensure the agent drive safely, comfortably, and efficiently.

- **DNN** [36]. The authors proposed a deep neural network-based (DNN) velocity control model to control the velocity of the vehicle in a human way.
- **Human**. Human driving data in real-world traffic scenes.

In addition to the baseline methods, we also consider several variations of our model.

- **L-DDPG**. Our proposed model without all the attention mechanisms.
- **LV-DDPG**. Our proposed model with the attention mechanism of relative velocity, but without the attention mechanism of relative distance.
- **LD-DDPG**. Our proposed model with the attention mechanism of relative distance, but without the attention mechanism of relative velocity.
- **A-DDPG**. The LADDPG model without LSTM.

We use the open-source code to reproduce the MPC-ACC and the DDPG model. We try our best to reproduce the DNN model based on the paper.

Evaluation Metrics: We design two series of metrics from two aspects: evaluating the ability for alleviating the chain reaction and evaluating the performance of the controlled vehicle. Two series of metrics are used in simulation experiments and real-world experiments respectively.

In the simulation experiments, in order to evaluate whether the model can alleviate the chain reaction caused by harsh driving behavior and improve the traffic capacity, we mainly designed three metrics: Affected Length (AL), Recovery Time (RT), Average Delay Index (ADI) [1] and Average Jerk (AJerk), as follows:

- **AL** refers to the number of rear vehicles affected by the HDB event. In the rear fleet of the HDB event, we record the velocity change rate of the vehicles . When the average velocity change rate is greater than 20% in continuous time steps, we believe that the vehicle has been affected, and the number of affected vehicles is the affected length. The velocity change rate is computed as $|V_{A_i}^{t+1} - V_{A_i}^{t}|/V_{A_i}^{t}$.
- **RT** refers to the time required for the affected vehicles to return to the normal driving state (i.e., the state before HDB occurring) after HDB occurring.
- **DI** refers to the delays in actual travel time compared to optimal travel time, and can be computed as $DI_{A_i} = T_{A_i}/T_{A_i}^{opt}$, where T_{A_i} is the duration of the vehicle A_i in a episode, and $T_{A_i}^{opt}$ is the optimal transport travel time of A_i [19]. The average delay index (ADI) overall vehicles on the road can be computed as $ADI = \frac{1}{I}\sum_{i \in I} DI_{A_i}$.

Note that, for the above metrics, the smaller the value is, the better the model can alleviate the chain reaction.

In the real-world data set experiments, we mainly consider three metrics to evaluate the safety, efficiency, and comfortability of the controlled vehicle, which are TTC, Mean THW (MTHW), and Average Jerk (AJerk).

- **TTC** is defined in Section 2. When the TTC is greater than a fixed value [35], the vehicle is driven safely.

Table 3: Evaluation of one autonomous vehicle

HDB Events	Metrics	Methods			
		DDPG	DNN	MPC-ACC	LADDPG
Hard braking	AL	15	16	14	**10**
	RT	8.81	10.21	9.12	**8.12**
	ADI	2.10	2.41	1.85	**1.68**
	AJerk	0.88	2.23	0.84	**0.72**
Hard acceleration	AL	13	15	13	**8**
	RT	15.8	18.01	15.39	**9.52**
	ADI	2.01	2.31	1.83	**1.66**
	AJerk	0.85	2.10	0.81	**0.69**

- **MTHW** is the mean THW of all the vehicles on the road. The smaller the MTHW is, the faster the vehicle drives. The MTHW is computed as:

$$MTHW = \frac{1}{\sum_{i \in I} i * T_{A_i}} \sum_{i \in I} \sum_{t \in T_{A_i}} THW_{A_i}^{t} \tag{14}$$

- **AJerk** is the average Jerk of all vehicles on the road. The smaller the AJerk is, the smoother the vehicle drives. The AJerk is calculated as:

$$AJerk = \sqrt{\frac{1}{\sum_{i \in I} i * T_{A_i}} \sum_{i \in I} \sum_{t \in T_{A_i}} (Jerk_{A_i}^{t})^2} \tag{15}$$

4.2 Performance on the Simulator

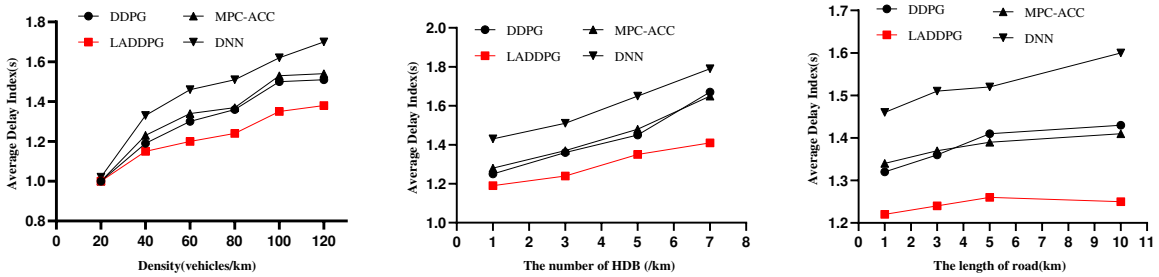
In this section, we prove the ability of the LADDPG for alleviating the chain reaction caused by harsh driving behaviors and improving the traffic capacity.

Environments Settings: We simulate two traffic environments based on the simulator. **1)** the first environment simulates a fixed one-lane road. The length of the road is $3km$, and the density of the vehicle is set as $100/km$, only one vehicle is controlled by using different methods (i.e. one autonomous vehicle). To better demonstrate the ability of our model to mitigate the chain reaction caused by HDBs, we add a hard braking event (decelerating continuously for 3 seconds at $-4m/s^2$ acceleration) and a hard acceleration event (accelerating continuously for 2 seconds at an acceleration of 4 m/s^2 without collision) as the HDBs to the vehicle in front of the autonomous vehicle, and we fixed the time of occurrence of the two events.

2) the second environment simulate a alterable one-lane road. We consider the different densities of the traffic (vehicles/km) and harsh-driving behaviors (HDBs/km) in the different lengths of roads. In addition, all the vehicles on the road are controlled by different models (i.e. all the vehicles are autonomous vehicles). We fixed the time of occurrence of the HDBs, and randomly assign an event, such as hard braking or acceleration, in each location of the HDB occurs.

Evaluation of One Autonomous Vehicle: In this environment, we divide the HDB events into hard braking and hard acceleration and separately study the AL, RT, DI, and $AJerk$ of the different baselines.

As shown in Table 3, we can see that all of the metrics of the LADDPG are smaller than the compared models. For the hard braking events, our model reduces the affected length by 28.6% to 37.5%



(a) The ADI of the models in different traffic densities

(b) The ADI of the models in different densities of HDB

(c) The ADI of the models on different lengths of road

Figure 4: Evaluation of multiple autonomous vehicles

and saves 7.8% – 20.5% recovery time. The average delay index and the average jerk of all vehicles are reduced by 9.2% – 30.1% and 14.2% – 67.7% respectively, which can prove the traffic capacity and the comfortability of the vehicles have been greatly improved by our LADDPG model. For the hard acceleration events, our model reduces 38.5% – 46.7% affected length and saves the recovery time by 38.1% to 47.1%. The average delay index of all vehicles is reduced by 9.3% to 28.1%, while the average jerk of all vehicles is the smallest of all compared methods.

We analyzed the reasons for the above results, when the vehicle occurs hard braking event, the traffic condition in front of it is relatively good. However, the compared methods can only observe the state of the first vehicle ahead. The autonomous vehicle controlled by baselines will brake as fast as the vehicle in front, and pass the brake-chain-reaction to the rear fleet, which increases the possibility of the 'phantom' traffic jam. The vehicle controlled by our LADDPG can observe the state of multiple vehicles ahead in historical time steps, which helps the vehicle to detect the harsh behaviors in the distance while judging the traffic condition is good or not. Therefore, our LADDPG acts decelerate in an adaptive way, reducing the pause time of the controlled vehicle, which can alleviate the chain reaction for the rear fleet.

Evaluation of Multiple Autonomous Vehicles: In this environment, we study the traffic capacity (average delay index of all vehicles) of different methods in variable traffic settings: traffic density, HDB density, and length of the road, as shown in Figure 4.

Firstly, we study different traffic densities (20, 40, 60, 80, 100, 120)$number\ of\ vehicles/km$ (see Figure 4a). Meanwhile, we fixed the length of the road and the number of HDB as $3km$ and 3 respectively. We can see that our model maintains the lowest delays in all settings, reducing 3.4% – 18.8% delay indexes. Secondly, we study the different densities of HDBs $(1, 3, 5, 7)/km$ (see Figure 4b). We fixed the length of the road and the traffic density as $3km$ and 80 respectively. Naturally, the more HDB events happen on the road, the lower the traffic capacity is. As the HDB density increases, the delay index of our model rises the slowest and is 4.8% – 21.2% lower than other methods. Thirdly, we study the different lengths of roads (1 km, 3km, 5km, and 10km), and fixed the traffic density and the HDB density as 80 and 3 respectively. As shown in Figure 4c, our method maintains the lowest delay indexes in all lengths of the road.

Therefore, the three results above prove that our LADDPG can effectively improve the traffic capacity in a one-lane road. The first reason is that LADDPG can observe the dynamic changes of the vehicles ahead, and make moderate decisions, reducing the impact on the rear vehicles. In addition, the vehicles controlled by our model can well cope with harsh driving behaviors such as hard braking and acceleration, to alleviate the chain reaction and avoid the 'phantom' traffic jam.

4.3 Performance on Real-World Data

In this section, we first compare our method with the state-of-the-art methods on the real-world dataset. Then we evaluate the effect of the components of attention and LSTM used in our model. Finally, we analyze the hyper-parameters setting of our spatial-temporal graph and reward function.

Our Model VS Baselines: We compare our velocity control model LADDPG against baselines by analyzing their efficiency, comfortability, and safety. Table 4 shows the MTHW (efficiency) and AJerk (comfortability) of the compared models, while Figure 5 shows their cumulative distribution functions of the TTC (safety).

In terms of efficiency, we can see that our model has the lowest mean value of THW, indicating that the autonomous vehicle controlled by our model can travel fast. For the comfortability of the autonomous vehicle, our model has the lowest average jerk, proving that the controlled vehicle can drive smoothly. In terms of safety, following the previous study [35], if the TTC of the current vehicle is greater than the predefined threshold (i.e. 1.5), the status of the vehicle is safe. Figure 5 shows the cumulative distributions of TTC for different methods on the real-world dataset. We can see that the probability that TTC is less than the threshold is almost 0%, that is, all the models can ensure the safety of the controlled vehicles and efficiently avoid collisions.

In short, the above results prove that our vehicles can drive as smoothly and fast as possible while ensuring safety.

Effect of Attention Mechanism: The attention mechanisms play a vital role in capturing the different importance of the vehicles ahead. To study the effectiveness of the attention, we conduct the ablation study by comparing the efficiency and comfortability of L-DDPG, LS-DDPG, and LD-DDPG with our complete model LADDPG. As shown in Table 4, the three models perform worse than the LADDPG model in terms of MTHW and AJerk. Specifically, after

Table 4: The performance of baselines and ours (LADDPG & variants) on real-world dataset

Metrics	Baselines				Ours				
	DDPG	DNN	MPC-ACC	HUMAN	**LADDPG**	L-DDPG	LV-DDPG	LD-DDPG	A-DDPG
MTHW	1.63	1.65	1.79	1.58	**1.29**	1.32	1.34	1.31	1.31
AJerk	1.34	3.31	1.02	3.02	**0.81**	0.92	0.88	0.91	1.07

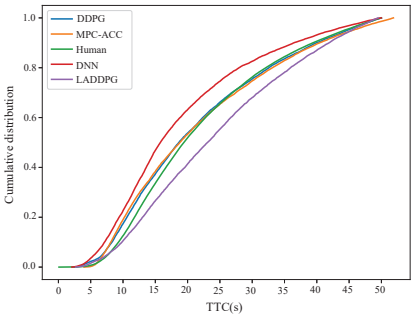


Figure 5: The cumulative distributions of TTC

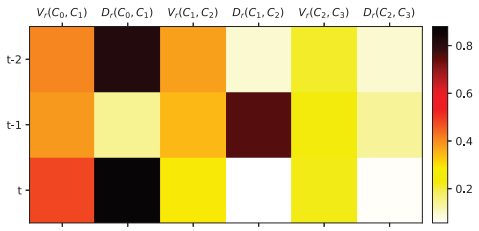


Figure 6: The heatmap of attention weights

deleting both VR-Att and DR-Att (i.e. L-DDPG) in the LADDPG, the MTHW increased by 2.2%, while the AJerk increased by 11.9%. After deleting the VR-Att (LD-DDPG), the MTHW and AJerk are increased by 1.6% and 12.3% respectively. After deleting the DR-Att (LV-DDPG), the MTHW and AJerk are increased by 3.9% and 8.6% respectively. Therefore, we prove that multi-global attention can efficiently learn the features of relative distance and velocity in the spatial-temporal graph.

To analyze the interpretability of the attention components, we visualize the importance scores outputted by VR-Att and DR-Att with respect to the spatial-temporal graph in Figure 6. In Figure 6, the x axes denote the relative velocity and distance between the adjacent vehicles, and the y axes denote the historical time steps. According to the heat map, the model learns to focus on a nearest vehicle (C_1) ahead of the autonomous vehicle(C_0), because the nearest vehicle will directly affect the autonomous vehicle, which deserves the higher weight. In contrast, the model assign the smaller weights for the vehicles further ahead, which is similar to human driving habits. The heat map proves that our VR-Att and DR-Att can adaptively focus on the important vehicles in front of the autonomous vehicle.

Effect of LSTM: The LSTM component is used to propagate the dynamic changes of We conduct the ablation study by comparing the

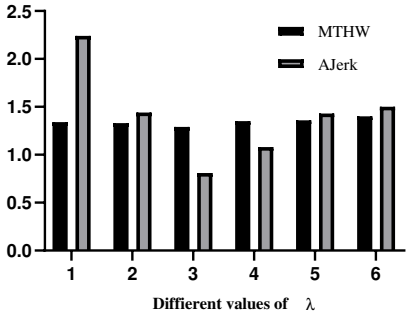


Figure 7: The effect of different time steps

efficiency and comfortability of A-DDPG with our complete LADDPG (see Table 4). Without the LSTM, we can see that the MTHW increases 1.5% while Ajerk increase 32.1%, which can demonstrate the effectiveness of the LSTM.

Effect of λ Time Steps and m Vehicles Ahead: In our study, the spatial-temporal graph is scalable, according to the number of vehicles ahead (m) and the length of historical time steps (λ). Following the previous work [15, 28], we consider three vehicles ahead of the autonomous vehicle (i.e. m is set to 3). For the historical time steps, we study the effectiveness of the model with different λ. As shown in Figure 7, when λ equals to 3, both the MTHW and AJerk are lowest. Thus, $\lambda = 3$ is taken as the default.

5 CONCLUSION

This work takes the first step to solve the notorious 'phantom' traffic jam problem through velocity control. Promising experimental results on real-world dataset have demonstrated the effectiveness of our method and its usefulness in real-world applications. We believe this work will draw more attention from the autonomous driving community to design more sophisticated methods and real-world applications specifically targeting this problem. As a future work, we will improve the practicality of this work by extending our method to support multi-lane situations. Specifically, multi-lane 'phantom' traffic jams, caused by other HDB like sudden change of lane, are more common than one-lane cases. Our method (spatial-temporal graph + attention + LSTM) naturally has power the process and predict the behavior of vehicles in multi-lane traffics.

ACKNOWLEDGMENTS

This work is supported by NSFC (No. 61802054, 61972069, 61836007, 61832017), scientific research projects of Quzhou Science and Technology Bureau, Zhejiang Pro-vince (No.2020D010), Alibaba Innovation Research (AIR).

REFERENCES

[1] 2021. Delay index calculation. https://kddcup2021-citybrainchallenge.readthedocs.io/en/latest/city-brain-challenge.html.

[2] Jeffery Archer. 2005. *Indicators for traffic safety assessment and prediction and their application in micro-simulation modelling: A study of urban and suburban intersections.* Ph.D. Dissertation. KTH.

[3] Meng Chen, Yan Zhao, Yang Liu, Xiaohui Yu, and Kai Zheng. 2020. Modeling spatial trajectories with attribute representation learning. *TKDE* (2020).

[4] YUE CUI, HAO SUN, YAN ZHAO, HONGZHI YIN, and KAI ZHENG. 2021. Sequential-knowledge-aware Next POI Recommendation: A Meta-learning Approach. *TOIS* (2021).

[5] A Karim Daoudia and Najem Moussa. 2003. Numerical simulations of a three-lane traffic model using cellular automata. *Chinese journal of physics* 41, 6 (2003), 671–681.

[6] LC Davis. 2004. Effect of adaptive cruise control systems on traffic flow. *Physical Review E* 69, 6 (2004), 066110.

[7] Liwei Deng, Hao Sun, Rui Sun, Yan Zhao, and Han Su. 2020. Efficient and Effective Similar Subtrajectory Search: A Spatial-aware Comprehension Approach. *TIST* (2020).

[8] Morris R Flynn, Aslan R Kasimov, Jean-Christophe Nave, EE Rosales, and B Seibold. 2009. Traffic modeling-Phantom traffic jams and traveling jamitons. *Traffic* 8, 29 (2009), 2016.

[9] Denos C Gazis and Robert Herman. 1992. The moving and "phantom" bottlenecks. *Transportation Science* 26, 3 (1992), 223–229.

[10] Hong-Xia Ge, Peng-jun Zheng, Wei Wang, and Rong-Jun Cheng. 2015. The car following model considering traffic jerk. *Physica A: Statistical Mechanics and its Applications* 433 (2015), 274–278.

[11] FE Gunawan. 2021. The fuel consumption density due to phantom traffic jam. In *IOP Conference Series: Earth and Environmental Science*, Vol. 729. IOP Publishing, 012025.

[12] Gabriel Hartmann, Zvi Shiller, and Amos Azaria. 2019. Deep reinforcement learning for time optimal velocity control using prior knowledge. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 186–193.

[13] Christopher D Higgins, Matthias N Sweet, and Pavlos S Kanaroglou. 2018. All minutes are not equal: travel time and the effects of congestion on commute satisfaction in Canadian cities. *Transportation* 45, 5 (2018), 1249–1268.

[14] ID Jacobson, LG Richards, and AR Kuhlthau. 1980. Models of human comfort in vehicle environments. *HUMAN FACTORS IN TRANSPORT RESEARCH EDITED BY DJ OBORNE, JA LEVIS* 2 (1980).

[15] Sheng Jin and Dian-Hai Wang. 2012. Car following model and simulation considering front traffic situation. *Beijing Gongye Daxue Xuebao(Journal of Beijing University of Technology)* 38, 8 (2012), 1236–1241.

[16] Vijay Gopal Kovvali, Vassili Alexiadis, and Lin Zhang PE. 2007. *Video-based vehicle trajectory data collection.* Technical Report.

[17] Douglas A Kurtze and Daniel C Hong. 1995. Traffic jams, granular flow, and soliton selection. *Physical Review E* 52, 1 (1995), 218.

[18] Stéphanie Lefevre, Ashwin Carvalho, and Francesco Borrelli. 2015. A learning-based framework for velocity control in autonomous driving. *IEEE Transactions on Automation Science and Engineering* 13, 1 (2015), 32–42.

[19] Lee Vien Leong, Tuti Azmalia Azai, Wins Cott Goh, and Mohammed Bally Mahdi. 2020. The Development and Assessment of Free-Flow Speed Models under Heterogeneous Traffic in Facilitating Sustainable Inter Urban Multilane Highways. *Sustainability* 12, 8 (2020), 3445.

[20] Yuxi Li. 2017. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274* (2017).

[21] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

[22] Shuncheng Liu, Han Su, Yan Zhao, Kai Zeng, and Kai Zheng. 2021. Lane Change Scheduling for Autonomous Vehicle: A Prediction-and-Search Framework. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 3343–3353. https://doi.org/10.1145/3447548.3467072

[23] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[24] Kai Nagel and Michael Schreckenberg. 1992. A cellular automaton model for freeway traffic. *Journal de physique I* 2, 12 (1992), 2221–2229.

[25] Morten Monrad Pedersen and Peder Thusgaard Ruhoff. 2002. Entry ramps in the Nagel-Schreckenberg model. *Physical Review E* 65, 5 (2002), 056705.

[26] Marcus Rickert, Kai Nagel, Michael Schreckenberg, and Andreas Latour. 1996. Two lane traffic simulations using cellular automata. *Physica A: Statistical Mechanics and its Applications* 231, 4 (1996), 534–550.

[27] D Schrank. 2019. Urban Mobility Report-Texas A&M University," 2019 Urban Mobility Report, 2019.

[28] Di-hua Sun, Yong-fu Li, and Chuan Tian. 2010. Car-following model based on the information of multiple ahead & velocity difference. *Systems Engineering-Theory & Practice* 30, 7 (2010), 1326–1332.

[29] Taku Takahama and Daisuke Akasaka. 2018. Model predictive control approach to design practical adaptive cruise control for traffic jam. *International Journal of Automotive Engineering* 9, 3 (2018), 99–104.

[30] Alireza Talebpour and Hani S Mahmassani. 2016. Influence of connected and autonomous vehicles on traffic flow stability and throughput. *Transportation Research Part C: Emerging Technologies* 71 (2016), 143–163.

[31] Christian Thiemann, Martin Treiber, and Arne Kesting. 2008. Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data. *Transportation Research Record* 2088, 1 (2008), 90–101.

[32] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. 2000. Congested traffic states in empirical observations and microscopic simulations. *Physical review E* 62, 2 (2000), 1805.

[33] Martin Treiber and Arne Kesting. 2013. Microscopic calibration and validation of car-following models–a systematic approach. *Procedia-Social and Behavioral Sciences* 80 (2013), 922–939.

[34] Martin Treiber, Arne Kesting, and Dirk Helbing. 2006. Delays, inaccuracies and anticipation in microscopic traffic models. *Physica A: Statistical Mechanics and its Applications* 360, 1 (2006), 71–88.

[35] Katja Vogel. 2003. A comparison of headway and time to collision as safety indicators. *Accident analysis & prevention* 35, 3 (2003), 427–433.

[36] Xiao Wang, Rui Jiang, Li Li, Yilun Lin, Xinhu Zheng, and Fei-Yue Wang. 2017. Capturing car-following behaviors by deep learning. *IEEE Transactions on Intelligent Transportation Systems* 19, 3 (2017), 910–920.

[37] Lanhang Ye and Toshiyuki Yamamoto. 2018. Modeling connected and autonomous vehicles in heterogeneous traffic flow. *Physica A: Statistical Mechanics and its Applications* 490 (2018), 269–277.

[38] Guohui Zhang, Yinhai Wang, Heng Wei, and Yanyan Chen. 2007. Examining headway distribution models with urban freeway loop event data. *Transportation Research Record* 1999, 1 (2007), 141–149.

[39] Yi Zhang, Ping Sun, Yuhan Yin, Lin Lin, and Xuesong Wang. 2018. Human-like autonomous vehicle speed control by deep reinforcement learning with double Q-learning. In *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 1251–1256.

[40] Yan Zhao, Shuo Shang, Yu Wang, Bolong Zheng, Quoc Viet Hung Nguyen, and Kai Zheng. 2018. Rest: A reference-based framework for spatio-temporal trajectory compression. In *SIGKDD*. 2797–2806.

[41] Kai Zheng, Yan Zhao, Defu Lian, Bolong Zheng, Guanfeng Liu, and Xiaofang Zhou. 2019. Reference-based framework for spatio-temporal trajectory compression and query processing. *TKDE* 32, 11 (2019), 2227–2240.

[42] Hao Zhou, Yan Zhao, Junhua Fang, Xuanhao Chen, and Kai Zeng. 2019. Hybrid route recommendation with taxi and shared bicycles. *Distributed and Parallel Databases* (2019), 1–21.

[43] Meixin Zhu, Xuesong Wang, and Yinhai Wang. 2018. Human-like autonomous car-following model with deep reinforcement learning. *Transportation research part C: emerging technologies* 97 (2018), 348–368.

[44] Meixin Zhu, Yinhai Wang, Ziyuan Pu, Jingyun Hu, Xuesong Wang, and Ruimin Ke. 2020. Safe, efficient, and comfortable velocity control based on reinforcement learning for autonomous driving. *Transportation Research Part C: Emerging Technologies* 117 (2020), 102662.