Comfort-aware Lane Change Planning with Exit Strategy for Autonomous Vehicle

Shuncheng Liu, Xu Chen, Yan Zhao, Han Su, Xiaofang Zhou, *Fellow, IEEE*, and Kai Zheng*, *Senior Member, IEEE.*

***Abstract*—Automation in road vehicles is an emerging technology that has developed rapidly over the last decade. There have been many inter-disciplinary challenges posed on existing transportation infrastructure by autonomous vehicles. In this paper, we conduct an algorithmic study on when and how an autonomous vehicle should change its lane, which is a fundamental problem in vehicle automation field and root cause of most 'phantom' traffic jams. We propose a prediction-and-decision framework, called *Cheetah* (Change lane smart for autonomous vehicle), which aims to optimize the lane changing maneuvers of autonomous vehicle while minimizing its impact on surrounding vehicles. In the prediction phase, Cheetah learns the spatio-temporal dynamics from historical trajectories of surrounding vehicles with a deep model (GAS-LED model) and predict their corresponding actions in the near future. A global attention mechanism and state sharing strategy are also incorporated to achieve higher accuracy and better convergence efficiency. Then in the decision phase, Cheetah looks for optimal lane change maneuvers for the autonomous vehicle by taking into account a few factors such as speed, impact on other vehicles and safety issues. A tree-based adaptive beam search algorithm is designed to reduce the search space and improve accuracy. In order to make our framework applicable to more scenarios, we further propose an improved Cheetah (Cheetah$^+$) framework that makes the autonomous vehicle adapt for exiting a road and meet the requirement for driving comfort. Extensive experiments offer evidence that the proposed framework can advance the state of the art in terms of effectiveness and efficiency.**

***Index Terms*—Autonomous vehicle, Lane change maneuvers, Trajectory prediction**

I. Introduction

With rapid urbanization and the increasing number of cars on the roads, traffic congestion has become a pressing issue in major cities worldwide. The resulting gridlock not only leads to wasted fuel but also contributes to elevated levels of air pollution [1]. Adding to the frustration of drivers, unexplained traffic jams, commonly known as 'phantom' traffic jams, are a frequent occurrence in everyday scenarios. These jams tend to emerge in areas where traffic density is high, and even minor disruptions, such as forced lane changes or hard braking, can set off a chain reaction leading to substantial congestion [2]. Effectively preventing phantom traffic jams requires all vehicles to maintain appropriate gaps between each other and avoid engaging in improper lane changes or sudden braking. However, achieving such a level of coordination and precision is a daunting task for human drivers. The complexities of monitoring and adjusting driving behaviors in real-time, amidst the hustle and bustle of traffic, can make it exceedingly challenging, if not impossible, for human drivers to consistently adhere to these requirements.

With the rapid advancement of vehicle automation technology, it is anticipated that the future adoption of autonomous vehicles will contribute significantly to achieving the goal of efficient and smooth transportation. Studies have demonstrated that the utilization of Adaptive Cruise Control (ACC) in autonomous vehicles for car-following control, incorporating smooth acceleration and braking, enables these vehicles to maintain a consistent distance from the preceding vehicle, thereby mitigating the occurrence of phantom traffic jams [3], [4]. However, in the context of Lane Change Assistant (LCA) systems in autonomous vehicles, the existing algorithms primarily focus on ensuring the safety and comfort of lane change maneuvers [2], [5], [6]. Unfortunately, there has been limited investigation into the potential impact on other surrounding vehicles and the resulting traffic conditions. For instance, Tesla's manual for the Model S advises drivers against using the Auto Lane Change feature on city streets or roads with dynamic traffic conditions [7]. The reason behind this cautionary warning is attributed to the fact that solely relying on the autonomous vehicle's status to determine when and how to change lanes, without considering the driving behaviors of surrounding vehicles, can lead to more severe traffic congestion or even accidents [8]. The absence of a comprehensive understanding of the surrounding traffic dynamics may result in improper lane change decisions, disrupting the flow of vehicles and potentially causing hazardous situations. Our work is grounded on the fundamental assumption that efficient LCA systems in autonomous vehicles should not only ensure safe, smooth, and efficient lane changes but also minimize their impact on surrounding vehicles. This necessitates the consideration of the driving behaviors of neighboring vehicles and the broader traffic conditions, facilitating a more synchronized and harmonious flow of traffic. By addressing this critical aspect, we aim to enhance the overall effectiveness and safety of autonomous vehicle operations.

To sum up, hard braking and improper lane changes are the two main reasons for most phantom traffic jams. In

S. Liu and X. Chen are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China. E-mail: {liushuncheng, xuchen}@std.uestc.edu.cn.
Y. Zhao is with the Department of Computer Science, Aalborg University, Aalborg 9220, Denmark. E-mail: yanz@cs.aau.dk.
H. Su and K. Zheng are with the Yangtze Delta Region Institute (Quzhou), School of Computer Science and Engineering, Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, China. E-mail: {hansu, zhengkai}@uestc.edu.cn. *K. Zheng is the corresponding author of the paper.
X. Zhou is with the Hong Kong University of Science and Technology, Hong Kong, China. E-mail: zxf@cse.ust.hk.

the field of autonomous driving, the first issue was studied more extensively and handled by ACC systems. However, the second issue has not been addressed well due to a few challenges: 1) lack of proper framework; 2) uncertainty of surrounding vehicles' driving behaviors; and 3) variety of vehicle maneuvers. In this work, we aim to address the above challenges and propose a new LCA system that fulfills SAE (Society of Automotive Engineers) level 4 or 5 automation [9].

For the simplicity of study, we consider a traffic scenario where there is one autonomous vehicle A and many conventional vehicles (with human drivers) $\mathbb{C}$ traveling on a multi-lane road. Our system can only control the behaviors of the autonomous vehicle: lane change, speed-up, speed-down, maintain speed. For other conventional vehicles, we can collect their real-time locations via Internet of Vehicles (IoV) or the autonomous vehicle's sensors. Our objective is to maximize the average speed of the autonomous vehicle without disturbing the other conventional vehicles as much as possible. To this end, we proposed a novel prediction-and-decision framework, called Cheetah, which consists of two phases. In the prediction phase, a trajectory prediction model, namely GAS-LED model (Global Attention and State sharing based LSTM Encoder-Decoder model), is proposed to model the dependencies of spatial and temporal features from neighboring vehicle's historical trajectories and forecast their future behaviors. The state sharing technique coordinated with global attention is designed to connect the encoder and the decoder, which not only achieves high accuracy but also reduces training time. Then during the decision phase, Cheetah hunts for optimal lane change maneuvers efficiently with a maneuver tree-based adaptive beam search algorithm. Instead of setting the width parameter heuristically, our search algorithm can automatically select high-confidence nodes and prune low-confidence nodes in each layer of the tree.

Though our previous work [10] has already achieved the objective of maximizing the average speed of the autonomous vehicle while minimizing its impact on surrounding conventional vehicles, there is still room of improving Cheetah in the following two aspects. (1) Exit strategy: when the autonomous vehicle needs to exit a road, it should enter the leftmost or rightmost lane in advance to make the exit process as smooth as possible. However, as Cheetah does not consider exit strategy in its framework, it may change lane suddenly when the target exit is approaching quickly, which can cause serious traffic jams or even accidents. (2) Comfort-awareness: to make the driving more comfortable, it requires the autonomous vehicle to reduce the fluctuations of physical movements [11]. However, at current stage Cheetah ignores the factors for driving comfort, and it is difficult to consider the discomfort caused by both longitudinal acceleration and lateral lane change. To tackle the above problems, we present Cheetah$^+$, a more advanced version of Cheetah with the following three improvements. (1) We first propose an Automated Exiting (AutoExit) strategy with an exit factor to consciously control the autonomous vehicle to enter the target lane (i.e., the leftmost or rightmost lane) at a distance from the target exit. (2) Besides, we design a discomfort factor that can measure the fluctuations of both longitudinal acceleration and lateral lane change of the autonomous vehicle. (3) Furthermore, maintaining the design principle of Cheetah, we reconstruct a set of improved search objectives by introducing an exit and a discomfort factor into our original objective. As a result, Cheetah$^+$ can not only maximize the average speed of the autonomous vehicle while minimizing its impact on surrounding conventional vehicles, but also make the exit more safely and the driving more comfortably.

The major value-added extension over our preliminary work [10] can be summarized as follows:

• We propose an Automated Exiting (AutoExit) strategy as well as an exit factor to make the autonomous vehicle adapt for exiting a road with the minimal forced lane change.
• We propose a discomfort factor to measure the fluctuations of both longitudinal acceleration and lateral lane change of the autonomous vehicle.
• Based on the well-designed exit and the discomfort factors, we present a set of improved search objectives w.r.t. different conditions of the autonomous vehicle.
• We conduct extensive experiments to evaluate Cheetah$^+$ on real and simulated data, verifying the effectiveness on multiple metrics.

II. Problem Statement

A. Problem Setting

The existing LCS systems mainly focus on the safety and comfortableness of driving, but ignore the impact of the lane change maneuvers of the autonomous vehicle on its surrounding vehicles, which may cause traffic jams. In this work, we try to reduce this impact as much as possible. More formally speaking, **our objective is to maximize the average speed of the autonomous vehicle while minimizing its impact on surrounding conventional vehicles.**

In this study, we consider an interactive environment where there are one autonomous vehicle A and a set of conventional vehicles $\mathbb{C}$ driving on a straight multi-lane road. For the sake of simplicity, parking and turning are not considered for now. The autonomous vehicle can obtain the real-time location of surrounding vehicles through its sensors or IoV (Internet of Vehicles), and make decision on lane change maneuver at each time instant within a time duration of interest $\mathbb{T}$. Without loss of generality, we assume the dimensions and performance of all vehicles are the same.

In the scenario mentioned above, we first define the notion of **lane**. A lane L is part of the road used to guide vehicles in the same direction. Usually, a road has multiple (at least two) lanes. Herein, all the lanes are numbered incrementally from the leftmost side to the rightmost side, i.e., $L_1, L_2, \cdots, L_\ell$, where L_1 and L_ℓ indicate the leftmost lane and the rightmost lane, respectively. An advantage of using this type of lane-level coordination system is to allow us to focus on the lane change behavior itself without worrying about the lateral position of the vehicle. Next, we introduce two basic units for spatial and temporal dimensions, respectively.

Trajectory Point. A trajectory point p indicates the location of a vehicle in a 2-dimensional space, wherein $p.L$ denotes the lateral lane number and $p.D_{lon}$ refers to the longitudinal

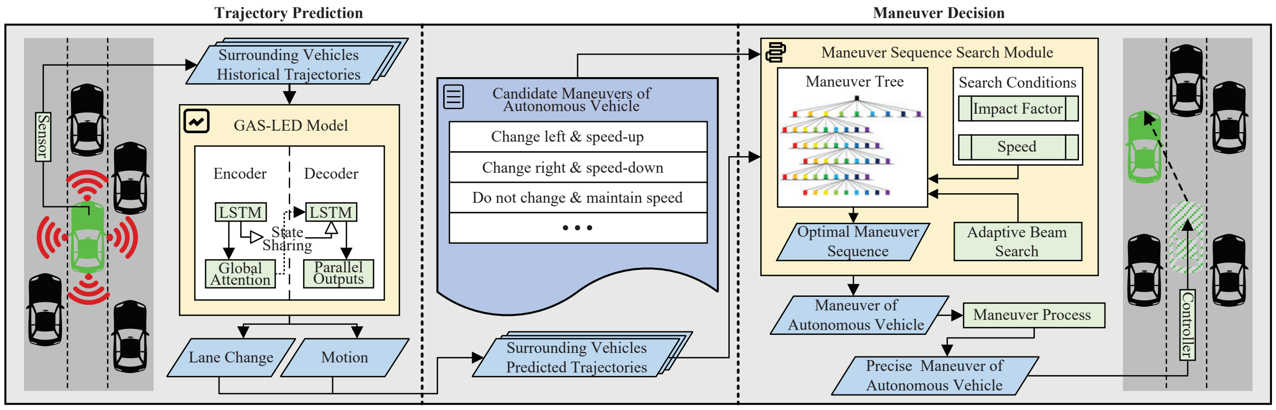

Trajectory Prediction
Maneuver Decision
Sensor
Surrounding Vehicles Historical Trajectories
GAS-LED Model
Encoder
Decoder
LSTM
LSTM
State Sharing
Global Attention
Parallel Outputs
Lane Change
Motion
Candidate Maneuvers of Autonomous Vehicle
Change left & speed-up
Change right & speed-down
Do not change & maintain speed
• • •
Surrounding Vehicles Predicted Trajectories
Maneuver Sequence Search Module
Maneuver Tree
Search Conditions
Impact Factor
Speed
Optimal Maneuver Sequence
Adaptive Beam Search
Maneuver of Autonomous Vehicle
Maneuver Process
Precise Maneuver of Autonomous Vehicle
Controller

Fig. 1. Cheetah overview

distance of the vehicle traveled from the starting point of a lane, which is a fixed point denoted as p_s i.e., $p_s.D_{lon}$=0. $p^t_{C_i}$ or p^t_A is used to denote the trajectory point of the vehicle C_i or A at time instant t. The $d(p^t_{C_i}, p^t_{C_j})$ denotes the longitudinal distance between two trajectory points, which can be calculated as follows:

$$d(p^t_{C_i}, p^t_{C_j}) = |p^t_{C_i}.D_{lon} - p^t_{C_j}.D_{lon}| \tag{1}$$

where $p^t_{C_i}.D_{lon}$ is the longitudinal distance of vehicle C_i at time instant t, which equals to $d(p_s, p^t_{C_i})$.

Time Step. In order to model the problem more concisely, we treat the continuous time duration as a set of discrete time steps, i.e., $\mathbb{T} = \{1, 2, \cdots, N\}$, and a time step $t \in \mathbb{T}$. It serves as the minimum frequency for the autonomous vehicle to make lane change decisions. Now the question is how to choose a suitable time granularity in real application scenarios? According to [12], the average duration of single-lane change is 4.6 seconds for a vehicle, which includes the transition time and the observation time required by the driver. In the field of microscopic traffic simulation [13], [14]. However, the transition and observation time of lane change is often ignored and the duration of single-lane change is considered less than 1.2 seconds. Therefore, without loss of generality, the time granularity in this work is set to 0.5 seconds, which means the time interval between two consecutive time steps (e.g., t and $t + 1$) is 0.5 seconds.

Trajectory. A trajectory $\mathcal{T}$ consists of a sequence of vehicle's trajectory points, ordered by time step, i.e., $\mathcal{T} = \langle p^1, p^2, \cdots, p^t\rangle$. We use $\mathcal{T}_{C_i}$ and $\mathcal{T}_A$ to denote the trajectory of the vehicle C_i and A, respectively.

Lane Change Maneuver. A maneuver is a series of (almost) simultaneous behaviors performed by a vehicle in order to accomplish a task (e.g., change lane). Inspired by recent studies in microscopic traffic field, we represent a lane change maneuver at time step t as a pair of a lateral lane change behavior and a longitudinal motion behavior, i.e., $M^t = [B^t_{lc}, B^t_m]$. B^t_{lc} can be one of the three behaviors: *change left* (Lt), *change right* (Rt), and *do not change* (Nt), i.e., $B^t_{lc} \in \{Lt, Rt, Nt\}$. B^t_m indicates the longitudinal distance travelled by the vehicle between current and the next time step, i.e., $B^t_m = d(p^t, p^{t+1})$.

Given a time duration of interest $\mathbb{T}$, the autonomous vehicle will perform a sequence of lane change maneuvers, denoted by $\mathcal{M} = \langle M^1, M^2, \cdots, M^N\rangle$, with the objective to maximize its average speed and minimize the impact on other conventional vehicles. In this work, we use maneuver to stand for lane change maneuver whenever the context is clear.

Restrictions. We pose some traffic restrictions. (1) Speed limit: all lanes are subject to two speed limits: V_{min} and V_{max}. (2) Lane change restrictions: a vehicle can only change to adjacent lane at each time step. Besides, it should keep D_{lcs} distance at minimum with the preceding and tailing vehicles in the target lane. (3) Safe following distance: all vehicles in the same lane should keep a safe following distance D_{ss}.

B. Framework Overview

Figure 1 illustrates the workflow of the framework. Next we will briefly introduce each component separately.

Trajectory Prediction. In order to reduce the impact of the autonomous vehicle on its surrounding conventional vehicles, Cheetah needs to predict the future trajectories of the surrounding conventional vehicles. To this end, we develop an LSTM-based deep model (i.e., GAS-LED model) to predict the trajectories of z time steps ahead by utilizing historical trajectories in the past n time steps. Instead of predicting the actual coordinates of each trajectory point directly, GAS-LED model is designed to make dual prediction for lateral lane change behavior B_{lc} and longitudinal motion behavior B_m of conventional vehicles at future time steps. For B_{lc}, the model estimates the probability of three behaviors and selects the behavior with the maximum probability as the predicted result. For B_m, the model outputs a longitudinal travel distance.

The most challenging part is how to maintain high accuracy in dynamic traffic environments, where the context of the autonomous vehicle keeps changing. To this end, we adopt a parallel model architecture integrated with a global attention mechanism to improve the prediction accuracy. In addition, an encoder-and-decoder state sharing mechanism is enabled to improve the convergence efficiency, so that the model can get up-to-date more quickly when the environment changes.

Maneuver Decision. Once GAS-LED model outputs the predicted trajectories of the surrounding conventional vehicles, Cheetah can search for the optimal maneuver sequence for the autonomous vehicle by using a search module (i.e., maneuver sequence search module). However, since the longitudinal motion behaviors B_m is a continuous value, we need to discretize it first to make the search process feasible and then convert it back to continuous value during post-processing. There are two goals during the search process. The first one is to maximize the average speed of the autonomous vehicle, and the second one is to minimize the impact on surrounding conventional vehicles, which is characterized by an impact factor to measure the extent to which the conventional vehicles are affected by the maneuver of the autonomous vehicle.

The most challenging part is how to reduce search space and achieve high efficiency. To this end, we propose to use a maneuver tree structure to represent the entire search space. Using our adaptive beam search algorithm, the maneuver tree can be kept within a tractable size so the search process can be performed efficiently.

III. Trajectory Prediction

To reduce the impact of the autonomous vehicle on its surrounding conventional vehicles, it is necessary for the autonomous vehicle to predict the future trajectories of its surrounding conventional vehicles. In Cheetah, we propose a novel trajectory prediction model (i.e., GAS-LED model). It divides the trajectory prediction task into two sub-tasks, namely lane change classification and motion regression, which can assure the accuracy of lane-level trajectory prediction and the reliability of maneuver decision-making. Moreover, GAS-LED model is designed to be trained efficiently with fast convergence so that the model can get up-to-date quickly when the environment changes dramatically.

A. Limitations of Existing Methods

Existing studies on trajectory prediction can be broadly classified into two categories, i.e., rule-based and learning-based trajectory prediction. In the sequel, we will briefly discuss the limitations of both categories of methods.

Rule-based Trajectory Prediction. Rule-based trajectory prediction methods mainly apply the transportation rules to simulate traffic flow models. For example, a vehicle will move to the right lane when there is no vehicle on its right front, or it will speed up when there is no preceding vehicle. The simulated models can be applied to predict the future possible actions of vehicles according to the real-time road environment at each time step. The cellular automaton algorithms are capable of simulating the traffic flows [15]–[17], which complete the task efficiently in simple scenarios like one-way straight lane and coarse-grained scene. However, these methods ignore the historical trajectories of vehicles, which makes them difficult to perform long-term predictions. Besides, since these methods are designed for simple traffic conditions, the level of accuracy will be reduced significantly in more complex scenarios such as vehicles on a multi-lane road in our studied problem.

Learning-based Trajectory Prediction. In recent years, with a success achieved in applying RNN to model non-linear temporal dependencies in sequence learning tasks, there have been plenty of works [18]–[23] utilizing RNN to predict the trajectories of vehicles. Overall, they have some common characteristics. (1) RNNs are adopted to extract long-term historical features. (2) The scale of the neural networks is large, and there are many training parameters and hyperparameters involved. (3) Single models output lateral and longitudinal information, which are continuous values in the Cartesian coordinate system.

These trajectory prediction models are able to predict long-term trajectories, and deal with the scenes with fine granularity and multi-interaction conditions based on the historical trajectories of vehicles. However, the training process of the aforementioned models is usually time-consuming due to a large number of parameters and hyperparameters. This will hinder the autonomous vehicle from updating model parameters promptly. Moreover, using a single model to predict both the lateral and longitudinal information may compromise the accuracy of lane-level trajectory prediction, leading to unreliable maneuver decisions. In general traffic scenarios, such as highways or urban roads, the road structure predominantly consists of multi-lane roads, where vehicles exhibit stable driving behavior at high speeds. Directly predicting the precise position of a vehicle in such scenarios can be costly in terms of training and prone to errors. This is due to the significant variation in the longitudinal driving distance compared to the relatively minor changes in lateral position. To simplify the prediction process while ensuring validity, the lane-level coordinate system can be utilized, as vehicles cannot drive side by side within the same lane due to lane width restrictions. By replacing coordinate predictions with predictions of lateral lane change behavior and longitudinal travel distance, the effectiveness of predictions can be improved while reducing model complexity. In summary, our goal is to develop an accurate and efficient lane-level trajectory prediction model.

B. GAS-LED Model

We propose a Global Attention and State sharing based LSTM Encoder-Decoder model, called GAS-LED model, for lane-level trajectory prediction. More specifically, a global attention mechanism is applied to assign attention weights to the encoder hidden state vectors for reflecting their different importance while avoiding complicating the model unduly. Besides, in order to improve the convergence efficiency of the model, we design an encoder-and-decoder state sharing mechanism to reduce the workload of calculation. The introduction of the encoder-decoder structure offers several significant benefits, including the ability to learn end-to-end, capture complex dependencies, generate accurate predictions, adapt to different scenarios, and handle diverse input and output formats. Furthermore, when combined with the proposed state sharing mechanism, the encoder-decoder structure enables efficient convergence and enhances its suitability for multi-step lane change and motion prediction. Furthermore, we adopt a dual-model structure, i.e., two similar GAS-LED models operate in parallel to perform lane change classification and motion regression simultaneously. The two models share the same underlying structure, which are trained separately and optimized for their own tasks (i.e., lane change classification and motion regression) to improve accuracy.

Let $\mathcal{T}^h$ denote a historical trajectory and $\mathcal{T}^f$ denote a predicted future trajectory. The length of $\mathcal{T}^h$ and $\mathcal{T}^f$ are indicated

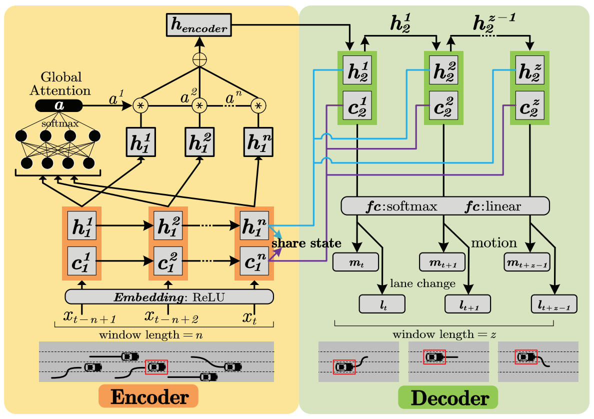


Fig. 2. GAS-LED model structure

by n and z, respectively, i.e., $\mathcal{T}^h = \langle p^{t-n+1}, p^{t-n+2}, \cdots, p^t \rangle$ and $\mathcal{T}^f = \langle p^{t+1}, p^{t+2}, \cdots, p^{t+z} \rangle$, where t represents the current time step. At each time step, taking a historical trajectory $\mathcal{T}^h$ of the past n time steps as input, the GAS-LED model outputs the predicted trajectory $\mathcal{T}^f$ of the future z time steps. Next, we detail the GAS-LED model.

Input. The autonomous vehicle can get a set of 2-dimensional trajectory points from radar sensors including lateral lane number $p.L$ and longitudinal distance $p.D_{lon}$. GAS-LED model needs to use a sequence of historical trajectory points to forecast a sequence of future trajectory points. The input features X are $[x_{t-n+1}, x_{t-n+2}, \cdots, x_t]$ with the input window length n, where each x has 14 features: 2 for predicted vehicle (C_0) and 12 for its surrounding vehicles (i.e., $\mathbb{C}_{C_0} = \{C_1, C_2, C_3, C_4, C_5, C_6\}$). The predicted vehicle C_0 has two features: current lane number $p^t_{C_0}.L$ and longitudinal distance $p^t_{C_0}.D_{lon}$. For the surrounding vehicles $\mathbb{C}_{C_0}$, we choose them based on the previous work [19], which have the most effect on the predicted vehicle. Each of them has two features: current lane number $p^t_{C_q}.L$ and relative longitudinal distance from the predicted vehicle $d(p^t_{C_0}, p^t_{C_q})$, where $C_q \in \mathbb{C}_{C_0}$.

Output. In the task of lane-level trajectory prediction, we focus on which lane the vehicle is in. Thus, for the lane change behavior B_{lc}, the model should output the probability of each lane change behavior, and then select the behavior with the maximum probability as the predicted result. Specifically, $\hat{l}_t$ denotes a 3-dimensional probability vector $[\hat{l}^1_t, \hat{l}^2_t, \hat{l}^3_t]$ at time step t, where $\hat{l}^1_t$=$\mathcal{P}(Lt)$, $\hat{l}^2_t$=$\mathcal{P}(Rt)$ and $\hat{l}^3_t$=$\mathcal{P}(Nt)$. For the motion behavior B_m, the model should output the longitudinal travel distance $d(p^t_{C_0}, p^{t+1}_{C_0})$, denoted as $\hat{m}_t$.

As a result, for lane change classification, the output results $Y_{\hat{l}}$ are $[\hat{l}_t, \hat{l}_{t+1}, \cdots, \hat{l}_{t+z-1}]$ with the output window length z, which represent the probabilities of Lt, Rt, and Nt in near future time steps. For motion regression, the output results $Y_{\hat{m}}$ are $[\hat{m}_t, \hat{m}_{t+1}, \cdots, \hat{m}_{t+z-1}]$ with the output window length z, which indicate the longitude travel distance of the predicted vehicle in near future time steps.

Trajectory Prediction Objective. Given the current time step t, our model needs to minimize the loss function as follows:

$$Loss = \sum_{\tau=t}^{t+z-1} \sum_{i=1}^{3} -l^i_\tau \log(\hat{l}^i_\tau) + \frac{1}{z} \sum_{\tau=t}^{t+z-1} (m_\tau - \hat{m}_\tau)^2 \tag{2}$$

where l^i_τ and m_τ denote the true values of the probability and distance, respectively. This loss function is the sum of Cross-entropy loss and Mean Squared error.

Model Structure. As shown in Figure 2, the historical information of the predicted vehicle and its surrounding vehicles is taken as the input of our GAS-LED model. Then, two similar models are applied in parallel to obtain multiple outputs simultaneously, i.e., the softmax activation for lane change classification, and the linear activation for motion regression. In the encoder part, the global attention mechanism assigns different attention weights to the hidden state vectors, which can reflect their different importance [24]. Besides, the decoder uses the last hidden and cell state vector of the encoder directly, as indicated by the blue and purple lines in Figure 2.

To be specific, in the encoder part, the input X is firstly embedded into $\tilde{X}$ via the embedding layer, i.e., $\tilde{X} = \phi_{eb}(X; W_{eb})$, where ϕ_{eb} is the embedding function with ReLU activation, W_{eb} denotes the embedding parameters. Secondly, $\tilde{X}$ is fed into the encoder LSTM, and the LSTM calculates the hidden and cell state vectors for each step $s \in \{1, 2, \cdots, n\}$ as follows:

$$h^s_1, c^s_1 = LSTM_1(h^{s-1}_1, c^{s-1}_1, \tilde{x}_s; W_{l1}) \tag{3}$$

where h^{s-1}_1 and $c^{s_1-1}_1$ are the hidden and cell state vectors at the previous step $s-1$, $\tilde{x}_s$ denotes s-th vector of $\tilde{X}$, and h^0_1 and c^0_1 are set as the initial vectors according to [25]. Thirdly, the global attention mechanism uses the concatenated hidden state vectors to output the attention weights $\boldsymbol{a}$ which can be calculated as follows:

$$\boldsymbol{a} = \phi_{ga}(Concat(h^1_1, h^2_1, \cdots, h^n_1); W_{at}) \tag{4}$$

where ϕ_{ga} is the global attention function with softmax activation [26]. Finally, the output vector of the encoder part $h_{encoder}$ can be formulated as $\sum_{s=1}^{n} a^s \cdot h^s_1$, where a^s denotes s-th attention weight of $\boldsymbol{a}$, and $h_{encoder}$ is the aggregation of all hidden state vectors w.r.t. different attention weights.

In the decoder part, the decoder LSTM is firstly used to decode the output vector from the encoder. However, the LSTM uses state sharing mechanism, which means that the state vectors of each decoding step equal to the last hidden and cell state vectors from the encoder LSTM, i.e., h^n_1 and c^n_1. To be specific, the decoder LSTM calculates the hidden and cell state vectors for each step $s' \in \{1, 2, \cdots, z\}$ as follows:

$$h^{s'}_2, c^{s'}_2 = \begin{cases} LSTM_2(h^n_1, c^n_1, h_{encoder}; W_{l2}) & s' = 1 \\ LSTM_2(h^n_1, c^n_1, h^{s'-1}_2; W_{l2}) & otherwise \end{cases} \tag{5}$$

At the first step, the input vector of the LSTM is $h_{encoder}$, after which the input will be changed to the hidden state vector at the previous step. Finally, the concatenated hidden state vectors are input to the fully connected output layer. Since we have two parallel GAS-LED models, and their calculations are the same except for the outputs, here, we distinguish their fully connected output layers as follows:

$$\begin{aligned} Y_l &= \phi_{ol}(Concat(h^1_2, h^2_2, \cdots, h^z_2); W_{ol}) \\ Y_m &= \phi_{om}(Concat(h^1_2, h^2_2, \cdots, h^z_2); W_{om}) \end{aligned} \tag{6}$$

where ϕ_{ol} is the output function (for lane change classification) with softmax activation, and ϕ_{om} is the output function (for motion regression) with linear activation.

To sum up, one GAS-LED model, used for lane change classification, outputs the lane change behaviors of the future time steps; another GAS-LED model, used for motion regression, outputs the longitudinal travel distance of the future time steps. We simply combine the two outputs to obtain the predicted trajectory points $(p.L, p.D_{lon})$ of the future time steps for the predicted vehicle C_0, as a result of its future trajectory $\mathcal{T}^f_{C_0}$.
Model Implementation. For the model implementation, the embedding layer contain 64 units with ReLU activation. The fully connected output layers for lane change classification and motion regression are comprised of 3 units and 1 unit, respectively. The encoder and decoder LSTMs have 128 units, while the global attention mechanism involves two fully connected layers (activated by softmax), each with 64 units.

IV. Maneuver Decision

After having the predicted future trajectories of the surrounding conventional vehicles, the autonomous vehicle needs to plan its maneuver based on the prediction. However, since the motion behavior B_m of the autonomous vehicle is a continuous value, performing a search against it directly will be extremely time-consuming. To speed up the search process, we develop two stages for maneuver decision: *maneuver sequence search* and *maneuver process*. In the maneuver sequence search stage, we search for a discretized version of the optimal maneuver sequence of the autonomous vehicle. Specifically, a discretized maneuver at time step t is a pair $\tilde{M}^t = [B^t_{lc}, \tilde{B}^t_m]$, where $\tilde{B}^t_m$ is a discretized version of B^t_m taking one of three behaviors: *speed-up* (Up), *speed-down* (Dw), and *maintain speed* (Mt), i.e., $\tilde{B}^t_m \in \{Up, Dw, Mt\}$. In the maneuver process stage, we generate the precise value for B^t_m based on $\tilde{B}^t_m$ obtained in the first stage. Since the logic of the maneuver process is easy to implement by using authoritative technology–Automotive Lane Change Aid (ALCA) [27], we mainly focus on the maneuver sequence search in this section.

A. Problem Definition

At time step t, given a prediction horizon z, the objective of maneuver sequence search is to find a maneuver sequence $\tilde{\mathcal{M}} = \langle \tilde{M}^t, \tilde{M}^{t+1}, \cdots, \tilde{M}^{t+z-1}\rangle$ of length z to 1) maximize the average speed of the autonomous vehicle (benefit); and to 2) minimize its impact on surrounding conventional vehicles (cost). In the following, we first quantitatively define the impact factor used in search condition 2), and then formally define the search objective.
Impact Factor F_{im}. At time step t, we assume the maneuver $\tilde{M}^t$ of the autonomous vehicle A can have impact on the set of conventional vehicles $\mathbb{C}_A$ within a radius of R. The impact, called impact factor, will emerge after the maneuver being performed, and thus it will be acquired at time step $t+1$. Formally, the impact factor F^{t+1}_{im} is defined as the sum of the sub-impact factor $f^{C_i,t+1}_{im}$ for each surrounding conventional vehicles $C_i \in \mathbb{C}_A$, i.e., $F^{t+1}_{im} = \sum f^{C_i,t+1}_{im}$. Specifically, we categorize the *impact situations* of maneuver within radius R into three types: queuing, jumping the queue, and crossing. Figure 3 illustrates the three situations, where the red shaded area, called conflicting area, is the location

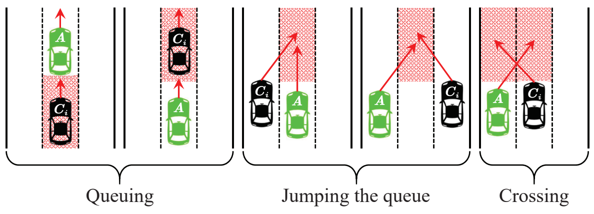


Fig. 3. Three impact situations

that both vehicles plan to arrive at the next time step. These three impacts encompass the effects of vehicle interactions on regular roads, including both the impacts of lateral lane changes and the impacts of longitudinal deceleration. Prior transportation studies have shown that crossing usually has the greatest impact, while queuing has the least [28], [29]. Therefore, by assigning sub-impact factors, we can effectively measure the severity of different impact situations, with emphasis on the relative differences between these factors. To distinguish the impacts of queuing, jumping the queue, and crossing, the absolute values of the three impact factors will be set as 1, 2, and 3, respectively. Next we present how to predict the three impact situations.

Obviously, the impact situations depend on the future trajectories of the surrounding conventional vehicles and the maneuver of the autonomous vehicle. A vehicle's state at time step t can be described using a pair $[p^t, p^{t+1}]$, which is used to describe its location at time step t and its immediate future location at time step $t+1$. Then the impact situations can be determined by checking if the autonomous vehicle and the conventional vehicle are on the same lane and their following distance is below the safe distance threshold. We use Algorithm 1 to predict the impact situations and calculate the impact factor F^{t+1}_{im}.

Algorithm 1: Impact factor calculation

```
Input: current time $t$, autonomous vehicle $A$, all conventional
       vehicles $\mathbb{C}_A$ in radius range $R$
Output: impact factor $F^{t+1}_{im}$
1  $F^{t+1}_{im} \leftarrow 0$;
2  foreach $C_i \in \mathbb{C}_A$ do
3      $f^{C_i,t+1}_{im} \leftarrow 0$;
4      if $p^{t+1}_{C_i}.L = p^{t+1}_A.L$ then
5          if $d(p^{t+1}_{C_i}, p^{t+1}_A) < D_{ss}$ then
6              if $p^t_{C_i}.L = p^t_A.L$ then
7                  $f^{C_i,t+1}_{im} \leftarrow 1$ ;          // queuing
8              else
9                  $f^{C_i,t+1}_{im} \leftarrow 2$ ;   // jumping the queue
10             end
11         end
12     else
13         if $p^{t+1}_{C_i}.L = p^t_A.L$ and $p^t_{C_i}.L = p^{t+1}_A.L$ then
14             if $d(p^{t+1}_{C_i}, p^{t+1}_A) < D_{lcs}$ or $d(p^t_{C_i}, p^t_A) < D_{lcs}$ then
15                 $f^{C_i,t+1}_{im} \leftarrow 3$ ;          // crossing
16             end
17         end
18     end
19     $F^{t+1}_{im} \leftarrow F^{t+1}_{im} + f^{C_i,t+1}_{im}$;
20 end
21 return $F^{t+1}_{im}$;
```

Search Objective. We define the overall search objective as a linear combination of the above two search conditions.

Specifically, at time step t, we aim to search for the maneuver sequence that maximizes the objective function defined below:

$$\arg\max_{\tilde{\mathcal{M}}} \sum_{\tau=t}^{t+z-1} V_A^{\tau+1} - F_{im}^{\tau+1} \tag{7}$$

Here, the speed of autonomous vehicle at time step $t+1$ (i.e., V_A^{t+1}) is calculated as $V_A^{t+1} = V_A^t + a_A^t * 0.5$, where a_A^t denotes the acceleration of the autonomous vehicle at time step t. Benefiting from the the discretized motion behavior $\tilde{B}_m^t$, a_A^t is determined by different $\tilde{B}_m^t$ as follows:

$$a_A^t = \begin{cases} a_{up} & \tilde{B}_m^t = Up \\ a_{dw} & \tilde{B}_m^t = Dw \\ 0 & \tilde{B}_m^t = Mt \end{cases} \tag{8}$$

where a_{up} and a_{dw} denote the *speed-up* acceleration and the *speed-down* acceleration, respectively. It should be noted that the range of features (V_A and F_{im}) is normalized to $[0, 1]$.

B. Maneuver Sequence Search Module

Maneuver Tree Structure. A maneuver tree is a tree-based data structure. The depth of the tree is z, and each non-leaf node has 9 child nodes (i.e., 3×3 pairs of discretized maneuver). Each node (except for the root) represents a kind of maneuver $\tilde{M}$ of the autonomous vehicle, and each edge holds two weights namely F_{im} (cost) and V_A (benefit) of choosing this edge. The root node is the current state while its child nodes represent the maneuvers.

Workflow Overview. At each time step, the maneuver tree will be reinitialized as a nine-complete tree with a fixed depth z, with the latest weights on each edge. First, based on the current trajectory point of the autonomous vehicle, some invalid nodes can be discarded based on the following rules:

- When the autonomous vehicle is in the rightmost lane, three direct child nodes (denoting the behavior of 'Change right') of the current node will be removed.
- When the autonomous vehicle is in the leftmost lane, three direct child nodes (representing the behavior of 'Change left') of the current node will be removed.

After deleting the invalid points, we use the adaptive beam search algorithm to search for the optimal path in the tree. Finally, the tree will be updated to prepare for the next process.

Adaptive Beam Search Algorithm. For a search process, a search algorithm is applied to search for a z-length path from the root node to the leaf node in the maneuver tree that achieves the search objective. Existing search algorithms use a predetermined number of candidate nodes in each layer (called the width) and only those nodes are expanded next, i.e., Brute-force search algorithm (width=9), Greedy search algorithm (width=1) [30] and Beam search algorithm (1<width<9) [31]. The greater the width, the fewer nodes are pruned and the more time-consuming the search process is. It is important to acknowledge that brute force search possesses the widest scope and can guarantee the optimal solution. However, as the depth of the search tree increases, the search space expands exponentially, leading to a substantial rise in search time. Once the depth reaches 10, the search response time exceeds 500 milliseconds, which surpasses the duration of a single time step (i.e., 0.5 seconds), rendering the search process unsuccessful. In fact, the width may depend on the weight distribution of each layer, and thus should be predetermined more adaptively. For instance, in one layer, the weights are (0.01, $\underline{0.4}$, $\underline{0.5}$, 0.003), in another layer, the weights are ($\underline{0.7}$, $\underline{0.6}$, $\underline{0.8}$, $\underline{0.9}$). If the algorithm fixes a small-sized width, it will ignore some valuable nodes (0.7,0.6), leading to a local optimum solution. Alternatively, if the algorithm fixes a large-sized width, it will consider excessive nodes (0.01,0.03). Thus, the width in each layer depends on the different number of nodes with high weights ($a.k.a.$ high-confidence).

Instead of fixing the search width, we adaptively select the candidate nodes with high confidence in each layer. To automatically select the candidate nodes, we define a threshold γ, which determines the minimum difference between high-confidence nodes and low-confidence nodes. Specifically, in i-th layer ($i \in \{1, 2, \cdots, z\}$), we sort all the nodes descending based on their cumulative weights $CW = \sum_{\tau=t}^{t+i-1} V_A^{\tau+i} - F_{im}^{\tau+i}$. Then we select the high-confidence nodes (from CW_{Top1} to CW_{TopK}) as the candidate nodes based on γ, i.e., $CW_{TopK} - CW_{TopK+1} \geq \gamma$. Thereafter, the nodes in i+1-th layer are the child nodes of the candidate nodes in i-th layer, while the non-candidate nodes are pruned to reduce search space. The search procedure in our adaptive beam search algorithm is similar to the classical beam search algorithm [31]. At each time step, once reaching the search depth (i.e., z), the algorithm will return a z-length path with the highest cumulative weights.

Example. As illustrated in Figure 4, the number of candidate nodes in each layer is different (4→6→7→5→5) following the threshold γ, and the non-candidate nodes are pruned accordingly. The search result is a 5-length path with the highest cumulative weights (yellow path).

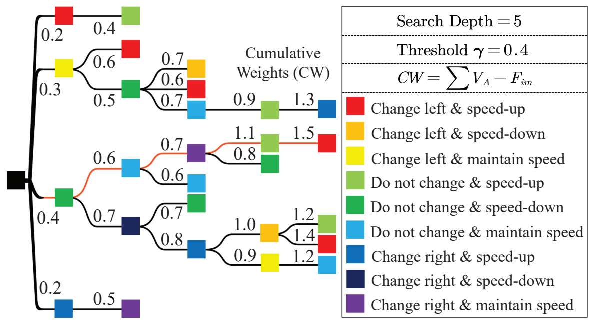

0.2
0.4
0.6
0.3
0.5
0.7
0.6
0.7
Cumulative Weights (CW)
0.9
1.3
1.1
1.5
0.8
0.6
0.6
0.4
0.7
0.7
0.8
1.0
1.2
1.4
0.9
1.2
0.2
0.5
Search Depth = 5
Threshold $\gamma = 0.4$
$CW = \sum V_A - F_{im}$
🟥 Change left & speed-up
🟧 Change left & speed-down
🟨 Change left & maintain speed
🟩 Do not change & speed-up
🟩 Do not change & speed-down
🟦 Do not change & maintain speed
🟦 Change right & speed-up
🟦 Change right & speed-down
🟪 Change right & maintain speed

Fig. 4. Example of adaptive beam search

V. Extension

As the extension of our previous work [10], we propose an improved Cheetah (Cheetah⁺) framework that makes the autonomous vehicle adapt for exiting a road and meet the requirement for driving comfort. Next, we first elaborate the two optimizations, and then present the improved search objectives used in Cheetah⁺.

A. Exit Strategy

A vehicle traveling on a road will always need to exit the road for reaching its destination. Usually, an autonomous

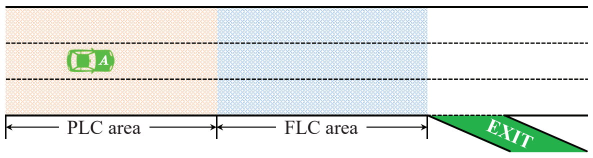


Fig. 5. PLC and FLC areas of AutoExit

vehicle should enter the leftmost or rightmost lane of a road in advance to ensure its exit. Intuitively, one can introduce a rule-based exiting strategy into a lane change algorithm, e.g., the Full Overlap Strategy (FOS) [32] that performs forced lane change within the predefined range in each lane. However, forced lane change before the target exit may cause serious traffic jams or even accidents since the autonomous vehicle ignores the impact of lane change maneuvers on its surrounding conventional vehicles.

To tackle the problem, we propose an Automated Exiting strategy, called AutoExit, to make the autonomous vehicle adapt for exiting a road and reduce forced lane change, when it needs to exit. The main idea behind AutoExit is to consciously control the autonomous vehicle to enter the target lane (i.e., the leftmost or rightmost lane) at a distance from the target exit. As shown in Figure 5, we define two areas before the target exit, i.e., *proactive lane-changing* (PLC) *area* and *forced lane-changing* (FLC) *area*. In PLC area, the autonomous vehicle needs to balance the benefit and cost of changing lane, and tries to enter or approach the target lane. In FLC area, the autonomous vehicle may perform forced lane change to ensure its exit, following FOS [32]. Apparently, if the autonomous vehicle enters or approaches the target lane in PLC area, the times of forced lane change will be greatly reduced. Therefore, we should design an additional factor that measures the benefit of the autonomous vehicle changing lane in PLC area, and introduce it into the original search objective in Equation (7). Next, we give the exit factor used in AutoExit.

Exit Factor F_{ex}. At time step t, we assume the maneuver $\tilde{M}^t$ of the autonomous vehicle A can have positive benefit when getting close to the target lane, or have zero benefit when staying away from the target lane. The benefit, called exit factor, will emerge after the maneuver being performed, and thus it will be acquired at time step $t+1$. The exit factor F_{ex}^{t+1} is determined by the target lane number (1 or ℓ), the current lane number $p_A^t.L$, and the current lane change behavior B_{lc}^t. Formally, when the target lane number is 1, i.e., the target lane is the leftmost lane L_1, the exit factor is defined as follows:

$$F_{ex}^{t+1} = \begin{cases} 1 & p_A^t.L \neq 1 \; and \; B_{lc}^t = Lt \\ 1 & p_A^t.L = 1 \; and \; B_{lc}^t = Nt \\ 0 & p_A^t.L \neq 1 \; and \; B_{lc}^t \neq Lt \end{cases} \tag{9}$$

where Lt denotes *change left*, and Nt denotes *do not change*. When the target lane number is ℓ, i.e., the target lane is the rightmost lane L_ℓ, the exit factor is defined as follows:

$$F_{ex}^{t+1} = \begin{cases} 1 & p_A^t.L \neq \ell \; and \; B_{lc}^t = Rt \\ 1 & p_A^t.L = \ell \; and \; B_{lc}^t = Nt \\ 0 & p_A^t.L \neq \ell \; and \; B_{lc}^t \neq Rt \end{cases} \tag{10}$$

where Rt denotes *change right*. The acquired exit factor enables AutoExit to measure the benefit of the autonomous vehicle changing lane in PLC area, and we can directly introduce it into the original search objective as an additional search condition (detailed in Section V-C). Once the autonomous vehicle leaves PLC area, i.e., enters FLC area, AutoExit utilizes FOS [32] to ensure that the autonomous vehicle can exit the road. We note that if the autonomous vehicle does not need to exit the road, AutoExit will be hidden as the default.

B. Comfort-aware Driving

In addition to taking speed and impact on other vehicles into account, autonomous driving needs to be comfortable and enjoyable to be accepted by passengers [33]. It requires the autonomous vehicle to minimize the avoidable fluctuations of physical movements [11]. Existing lane change algorithms [5], [6], [34], [35] enhance driving comfort of the autonomous vehicle by minimizing its Jerk feature. Jerk feature, defined as the change rate of acceleration, is used to measure driving comfort since it has a strong influence on the comfort of the passengers [36]. The larger the Jerk feature is, the more uncomfortable passengers will feel. However, the existing algorithms mainly focus on the change rate of longitudinal acceleration, ignoring the discomfort caused by frequent lateral lane change. Obviously, frequent lane change of the autonomous vehicle could cause passengers discomfort and even lead to potential accidents.

To solve the problem above, we design a discomfort factor that can measure the fluctuations of both longitudinal acceleration and lateral lane change of the autonomous vehicle, and introduce it into the original search objective in Equation (7), thus making the autonomous vehicle meet the requirement for driving comfort more comprehensively. Next, we elaborate the discomfort factor.

Discomfort Factor F_{di}. At time step t, we assume the maneuver $\tilde{M}^t$ of the autonomous vehicle A has influence on the driving comfort. The influence, called discomfort factor, will emerge after the maneuver being performed, and thus we assume it can be acquired at time step $t+1$. Formally, the discomfort factor F_{di}^{t+1} is defined as the sum of two sub-discomfort factors: 1) the change rate of longitudinal acceleration between time step $t-1$ and t, denoted by $f_{lonac}^{(t-1,t)}$; and 2) the change extent of lateral lane change behavior between time step $t-1$ and t, denoted by $f_{latlc}^{(t-1,t)}$. Overall, $F_{di}^{t+1} = f_{lonac}^{(t-1,t)} + f_{latlc}^{(t-1,t)}$. Based on the calculation of Jerk feature [11], the change rate of longitudinal acceleration between $t-1$ and t is calculated as follows:

$$f_{lonac}^{(t-1,t)} = \frac{|a_A^t - a_A^{t-1}|}{a_{up} - a_{dw}} \tag{11}$$

where a_A^t denotes the acceleration of the autonomous vehicle at time step t, and a_{up} and a_{dw} denote the *speed-up* acceleration and the *speed-down* acceleration, respectively. The range of $f_{lonac}^{(t-1,t)}$ is $[0, 1]$. When there is a noticeable change in the longitudinal acceleration of the autonomous vehicle between $t-1$ and t, the value of $f_{lonac}^{(t-1,t)}$ will be close to 1; otherwise, it will be close to 0. The change extent of lateral lane change behavior between $t-1$ and t is defined as follows:

$$f_{latlc}^{(t-1,t)} = \begin{cases} 1 & B_{lc}^{t-1} \neq Nt \; and \; B_{lc}^t \neq Nt \\ 0 & otherwise \end{cases} \tag{12}$$

where B_{lc}^{t-1} denotes the lane change behavior at time step $t-1$. The range of $f_{latlc}^{(t-1,t)}$ is $[0,1]$. When the autonomous vehicle exhibits frequent lateral lane changes between $t-1$ and t, the value of $f_{latlc}^{(t-1,t)}$ will be close to 1; otherwise, it will be close to 0. $f_{lonac}^{(t-1,t)}$ and $f_{latlc}^{(t-1,t)}$ work together to evaluate the comfort of the autonomous vehicle, aiming to reduce instances of frequent acceleration, deceleration, and lane-changing behavior. Thereafter, we can directly introduce the discomfort factor into the original search objective as an additional search condition (detailed in Section V-C).

C. Improved Search Objectives

Cheetah⁺ is an improved version of Cheetah, which is designed to maximize the average speed of the autonomous vehicle while minimizing its impact on surrounding conventional vehicles, as well as make the autonomous vehicle adapt for exiting a road and meet the requirement for driving comfort. Benefiting from the well-designed exit factor and discomfort factor, we can introduce them into the original search objective as the additional search conditions, and search for the optimal maneuver sequence by using the maneuver sequence search module described in Section IV-B. Specifically, the improved search objectives used in Cheetah⁺ are determined by different conditions of the autonomous vehicle as follows:

• **Condition 1: Do not need to exit / Need to exit before PLC area.** At time step t, if the autonomous vehicle does not need to exit the road or it needs to exit and travels before PLC area, we aim to search for its maneuver sequence that maximizes the objective function as follows:

$$\arg\max_{\tilde{\mathcal{M}}} \sum_{\tau=t}^{t+z-1} V_A^{\tau+1} - F_{im}^{\tau+1} - F_{di}^{\tau+1} \tag{13}$$

where $\tilde{\mathcal{M}}$ denotes the maneuver sequence of length z, and $V_A^{\tau+1}$, $F_{im}^{\tau+1}$ and $F_{di}^{\tau+1}$ denote the speed, the impact factor and the discomfort factor of the autonomous vehicle at time step $\tau+1$, respectively. In this condition, the autonomous vehicle need to maximize its average speed while minimizing its impact on surrounding conventional vehicles, as well as meet the requirement for driving comfort.

• **Condition 2: Need to exit in PLC area.** At time step t, if the autonomous vehicle needs to exit the road and travels in PLC area, we aim to search for its maneuver sequence that maximizes the objective function as follows:

$$\arg\max_{\tilde{\mathcal{M}}} \sum_{\tau=t}^{t+z-1} V_A^{\tau+1} - F_{im}^{\tau+1} + F_{ex}^{\tau+1} - F_{di}^{\tau+1} \tag{14}$$

where $F_{ex}^{\tau+1}$ denotes the exit factor of the autonomous vehicle at time step $\tau+1$. In this condition, the autonomous vehicle needs to maximize its average speed while minimizing its impact on surrounding conventional vehicles, as well as adapt for exiting the road and meet the requirement for driving comfort. For the above two conditions, the range of features (V_A, F_{im}, F_{ex}, and F_{di}) is normalized to $[0,1]$.

• **Condition 3: Need to exit in FLC area.** At time step t, if the autonomous vehicle needs to exit the road and travels in FLC area, the objective function is set to null, and the maneuver sequence search module is replaced with FOS [32] to ensure that the autonomous vehicle can exit the road successfully.

VI. Experiments

A. Experimental Settings

We implement all models and algorithms in Python on Linux, and run the experiments on a machine with an Intel(R) CPU i7-4770@3.4GHz and 32G RAM.

Datasets. As it requires interaction between the autonomous and conventional vehicles, most of the experiments are conducted in two simulated environments, called **SIM** and **SIM⁺**, where SIM is used for evaluating Cheetah and SIM⁺ is used for evaluating the improved Cheetah (i.e., Cheetah⁺). Specifically, SIM is generated using the microscopic traffic simulator [15], [35], [37], [38], which simulates the traffic behaviors of 600 conventional vehicles on a straight six-lane road of length $3km$ with periodic boundary conditions. We utilize Cheetah to control the autonomous vehicle to perform lane change maneuvers in SIM. SIM⁺ is generated based on SIM, we further set an exit at the $2.5km$ of the rightmost lane. We utilize Cheetah⁺ to control the autonomous vehicle to perform lane change maneuvers for exiting the road. For the sake of simplicity, only the autonomous vehicle is allowed to exit the road, and it will immediately enter the starting point of the rightmost lane after exiting. In both SIM and SIM⁺, we set the traffic restrictions as $V_{min} = 0km/h$, $V_{max} = 115km/h$, $D_{ss} = 10m$ and $D_{lcs} = 10m$ according to [39].

Furthermore, we evaluate our GAS-LED model on a dataset constructed by merging two real-world datasets: NGSIM US-101 [40] and I-80 [41]. The merged dataset, called **REAL**, consists of real trajectories of conventional vehicles traveling on a $1.14km$-length highway segment with six straight lanes. The original trajectories were captured at $10Hz$ (10 frames per second) over a period of 45 minutes. We preprocess the dataset by 1) removing the frames with less than 10 vehicles and 2) down-sampling the datasets to a rate of $2Hz$ (setting the time step to 0.5s). After preprocessing, REAL has $10,542$ frames with $9,864$ conventional vehicles. On average, there are 223 vehicles per frame, and the speed of vehicle is $33.4km/h$. Since the distributions of REAL, SIM and SIM⁺ are similar, we can train our GAS-LED model on REAL and use it to do the trajectory prediction in SIM and SIM⁺.

Parameters Settings. Unless otherwise specified, we set the parameters in Cheetah and Cheetah⁺ throughout the experiments as follows. For the *trajectory prediction* phase, we set both the length of input historical trajectories and the length of predicted trajectories to 5 (i.e., $n = z = 5$), following the settings used in the existing work [42]. In addition, we train our GAS-LED model by using the Adam optimizer [43] for 15 epochs with a learning rate of 0.001 and a batch size of 64, following the implementations of the previous work [10].

For the *maneuver decision* phase, we set the *speed-up* acceleration a_{up} to $1.2m/s^2$ and the *speed-down* acceleration a_{dw} to $-3m/s^2$, according to the rates recommended by [44]. We set the length of both PLC area and FLC area to $800m$ according to [32]. Moreover, we set the adaptive beam search threshold as $\gamma = 0.4$, and the impact factor radius as $R = 38m$, which are evaluated in Section VI-F.

Compared Methods. We compare Cheetah against several lane change algorithms, including STNS [34], FLS [35], and

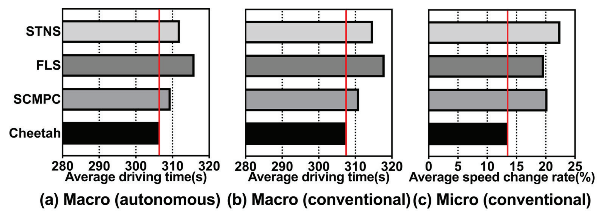


Fig. 6. Effectiveness of Cheetah

SCMPC [5]. In addition, we compare Cheetah⁺ against the modified versions of the above lane change algorithms, i.e., STNS*, FLS*, SCMPC*, and Cheetah*, which are adapted for exiting the road by directly introducing the rule-based exiting strategy FOS [32]. For the evaluation of trajectory prediction, we compare GAS-LED model against the state-of-the-art trajectory prediction models including LSTM [20], S-LSTM [21], CS-LSTM [42], and ED-LSTM [18]. For the evaluation of maneuver decision, we compare our adaptive beam (ABM) search algorithm against several search algorithms, i.e., Brute-force (BF) search algorithm with fixed width 9, Greedy (GD) search algorithm [30] with fixed width 1, and Beam (BM) search algorithm [31] with fixed width 4.

B. End-to-End Evaluation of Cheetah

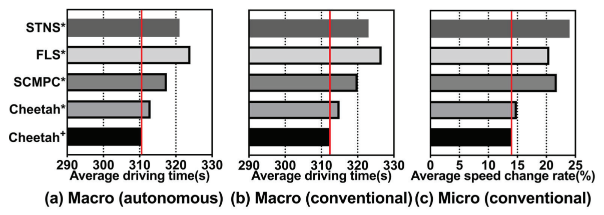
In this section, we study the end-to-end performance of Cheetah by comparing it against several lane change algorithms (STNS, FLS, and SCMPC). We conduct 100 tests in the simulated environment SIM, in each of which the autonomous vehicle is initialized at a random position and drives through $3km$, and measure the effectiveness from two aspects:

- Macroscopic: we record the end-to-end time of the autonomous vehicle and all conventional vehicles traveling $3km$.
- Microscopic: we record the average speed change rate of the conventional vehicles affected by (i.e., within radius $R = 38m$ of) the autonomous vehicle. The speed change rate is defined as $|V_{C_i}^t - V_{C_i}^{t-1}|/V_{C_i}^{t-1}(\%)$, where $V_{C_i}^t$ and $V_{C_i}^{t-1}$ are the speeds of a conventional vehicle C_i in two consecutive time steps. This metric directly reflects the extent of impact on the surrounding vehicles.

Macroscopic. We report the average end-to-end driving time of the autonomous vehicle and the conventional vehicles in Figures 6a and 6b. As shown, Cheetah achieves the shortest average driving time for both, which clearly demonstrates that Cheetah can maximize the average speed of the autonomous vehicle, while minimizing the impact on conventional vehicles.
Microscopic. We report the average speed change rate of the surrounding conventional vehicles in Figure 6c. We can see that Cheetah causes the least significant impact on the speeds of surrounding conventional vehicles, demonstrating the effectiveness from the perspective of microscopic traffic.

C. End-to-End Evaluation of Cheetah⁺

In this section, we study the end-to-end performance of Cheetah⁺ by comparing it against the modified versions of lane change algorithms (STNS*, FLS*, SCMPC*, and Cheetah*). We conduct 100 tests in the simulated environment SIM⁺, in each of which the autonomous vehicle is initialized at



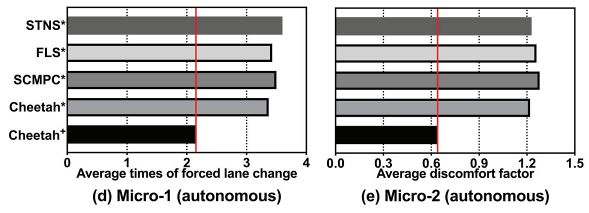


Fig. 7. Effectiveness of Cheetah⁺

a random position and drives through $3km$. In addition to measuring the effectiveness from the macroscopic and microscopic aspects as in Section VI-B, we further introduce two microscopic metrics to holistically evaluate the effectiveness of the autonomous vehicle in SIM⁺. The first metric (Micro-1) counts the times of forced lane change of the autonomous vehicle that occurs in FLC area, which reflects the possibility of causing traffic jams or accidents, and the second metric (Micro-2) measures the discomfort factor of the autonomous vehicle, which reflects the driving comfort.
Macroscopic. We report the average driving time of the autonomous vehicle and the conventional vehicles in Figures 7a and 7b. As expected, Cheetah⁺ achieves the shortest average driving time for both vehicles. This clearly demonstrates that Cheetah⁺ can maximize the average speed of the autonomous vehicle when it needs to exit the road, while minimizing the impact on its surrounding conventional vehicles.
Microscopic. We report the average speed change rate of the surrounding conventional vehicles in Figure 7c. We can see that Cheetah⁺ causes the least significant impact on the speeds of surrounding conventional vehicles, demonstrating the effectiveness of our framework from the perspective of microscopic traffic. Further, we give the average times of forced lane change of the autonomous vehicle that occurs in FLC area and the average discomfort factor of the autonomous vehicle in Figures 7d and 7e, respectively. As we can see, Cheetah⁺ achieves the least average times of forced lane change and the lowest average discomfort factor. This clearly demonstrates that 1) AutoExit with the exit factor can significantly reduce the occurrences of forced lane change when the autonomous vehicle exits the road; and 2) Cheetah⁺ with the discomfort factor can minimize the avoidable fluctuations of both longitudinal acceleration and lateral lane change.

D. Evaluation of Trajectory Prediction

In this section, we take a break-down evaluation of our GAS-LED model in trajectory prediction phase by comparing it with the state-of-the-art trajectory prediction models (LSTM, S-LSTM, CS-LSTM, and ED-LSTM). To this end, we split the merged dataset REAL into training and test sets with a splitting ratio of $4:1$. All the models share the same input and output structures, and they are trained on the training set and tested on the test set.

TABLE I
EFFECTIVENESS OF GAS-LED MODEL

Lane change classification ACC(%)					
Models	**Prediction time step**				
	1	**2**	**3**	**4**	**5**
LSTM	91.2	89.7	88.5	88.1	87.6
S-LSTM	94.5	93.1	92.8	92.2	90.5
CS-LSTM	95.1	94.3	93.9	93.2	91.8
ED-LSTM	95.8	95.2	94.6	93.5	92.3
GAS-LED	**96.1**	**95.7**	**94.8**	**94.2**	**93.5**

Motion regression MSE(ft)					
Models	**Prediction time step**				
	1	**2**	**3**	**4**	**5**
LSTM	0.60	0.89	1.39	1.58	1.87
S-LSTM	0.36	0.91	1.07	1.33	1.63
CS-LSTM	0.39	0.81	1.27	1.59	1.90
ED-LSTM	0.27	0.63	0.93	1.20	1.49
GAS-LED	**0.23**	**0.60**	**0.89**	**1.16**	**1.47**

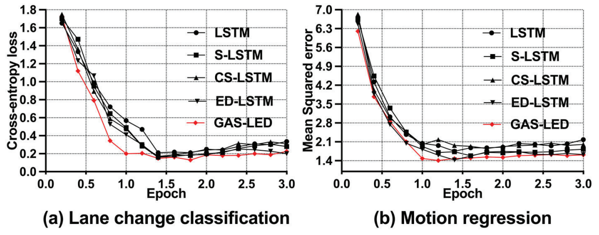


Fig. 8. Efficiency of GAS-LED model

Effectiveness of GAS-LED Model. We study the lane change classification task and the motion regression task separately, and for both tasks we use historical trajectories of length 5 to predict trajectories of the next 5 time steps on the test set. We report the Accuracy (ACC) of the lane change classification task and the Mean Squared error (MSE) of the motion regression task in Table I. As depicted, for both tasks, our proposed GAS-LED model outperforms all the other models for all prediction time steps.

Efficiency of GAS-LED Model. To study the efficiency of the model, we record the curve of the training loss in the first three training epochs for all the models on the training set. We choose the Cross-entropy loss and Mean Squared error as the loss metrics, and plot the loss curves in Figures 8a and 8b. As shown, our GAS-LED model is minimized at around 1 epoch, while the other models are minimized at around 1.5 epochs. This is because our model relies on the state sharing of encoder-decoder structure, thus reducing the training convergence time.

Necessity of a Continuously Updated Model. To demonstrate the necessity of a continuously updated model in maintaining a high trajectory prediction accuracy, we compare the effectiveness of GAS-LED model under two training strategies:

- Constant model (CT): we train GAS-LED model using the entire training set, and use it to do the trajectory prediction for the entire test set.
- Updated model (UD): we further divide the training and test sets into time windows. We train GAS-LED model on each training window, and use it to do the trajectory prediction on the corresponding test window.

For GAS-LED model under CT and UD, we record the average lane change ACC and motion MSE of all prediction time steps in Table II. We can see that the effect of the updated model is much better than that of the constant model. This proves that the trajectory prediction model needs to be updated in continuously changing environments to maintain accuracy.

TABLE II
EFFECTIVENESS OF GAS-LED MODEL UNDER CT AND UT STRATEGIES

Strategies	Lane change ACC(%)	Motion MSE(ft)
CT	86.5	1.94
UD	**92.7**	**1.49**

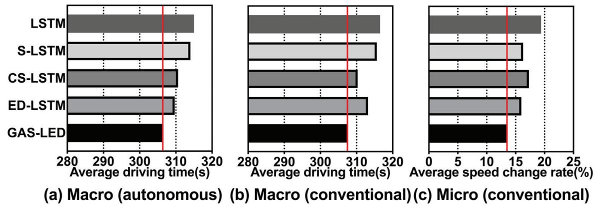


Fig. 9. Effectiveness of GAS-LED model in Cheetah

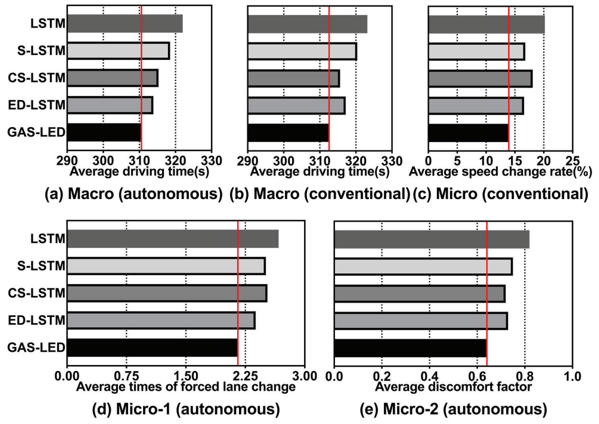


Fig. 10. Effectiveness of GAS-LED model in Cheetah$^+$

Effectiveness of GAS-LED Model in Cheetah and Cheetah$^+$. We study the effectiveness of GAS-LED model when putting it in Cheetah and Cheetah$^+$ with maneuver decision. We compare GAS-LED model against other trajectory prediction models in terms of effectiveness as in Sections VI-B and VI-C. The results are shown in Figures 9 and 10. We can see that in Cheetah and Cheetah$^+$ with the maneuver sequence search module, the proposed GAS-LED model outperforms all the competitive models, due to the synergy of our GAS-LED model and maneuver sequence search module. Other models pay more attention to location information rather than the lane change behaviors. This shows that our GAS-LED model not only outperforms other models in the stand-alone trajectory prediction task, but also is effective in lane change planning.

Ablation Study of GAS-LED Model. We further evaluate the performance of the dual-model structure (DMS) and the state sharing mechanism (SSM) proposed in GAS-LED model. The ablation study is conducted by comparing GAS-LED model against two alternative models: GAS-LED-w/o-DMS, which utilizes a single-model structure similar to multi-task learning, and GAS-LED-w/o-SSM, which uses a vanilla encoder-decoder approach [18] instead of SSM. Specifically, we record their average lane change ACC and motion MSE of all prediction time steps in Table III. We can see that GAS-LED model outperforms the GAS-LED-w/o-DMS and GAS-LED-w/o-SSM models. This provides strong evidence that both the DMS and SSM significantly enhance the effectiveness of lane change classification and motion regression.

TABLE III
ABLATION STUDY OF GAS-LED MODEL

Models	Lane change ACC(%)	Motion MSE(ft)
GAS-LED-w/o-DMS	88.3	1.77
GAS-LED-w/o-SSM	89.2	1.65
GAS-LED	**92.7**	**1.49**

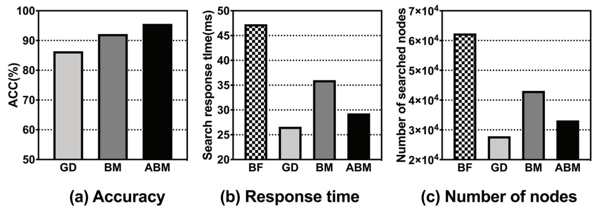


Fig. 11. Effectiveness and efficiency of adaptive beam search

E. Evaluation of Maneuver Decision

In this section, we take a break-down evaluation of our maneuver sequence search module in maneuver decision phase. We compare our adaptive beam (ABM) search algorithm against several search algorithms (BF, GD, and BM). We conduct 100 tests in SIM and SIM⁺ separately, in each of which the autonomous vehicle is initialized at a random position and drives through $3km$. Since the experimental results of all search algorithms are similar in both simulated environments, here, we only present the results in SIM.

Effectiveness of Adaptive Beam Search Algorithm. To study the effectiveness of the adaptive beam search algorithm adopted in our maneuver sequence search module, we compare the accuracy of the maneuver results of different search algorithms. As BF search algorithm without any pruning is guaranteed to produce the globally optimal result, we define the accuracies of GD, BM, and ABM as the ratios that these algorithms will generate the same maneuver as BF. We report the accuracy results in Figure 11a. We can see that the accuracy of our ABM search algorithm is noticeably better than other algorithms since we take adaptive nodes in each layer of the maneuver tree to search the optimal path. ABM search algorithm improves the accuracy by $3.6 \sim 10\%$ compared to other algorithms, which evidences the effectiveness of our adaptive beam search algorithm.

Efficiency of Adaptive Beam Search Algorithm. We also compare the search response time and the number of searched nodes of different search algorithms in Figures 11b and 11c. As we can see, ABM search algorithm prunes $23 \sim 46\%$ nodes during the search process, which results in $18 \sim 38\%$ response time reduction for BM search algorithm and BF search algorithm. GD search algorithm takes the shortest response time since it only considers one candidate node in each layer. Therefore, it will mistakenly prune many high-confidence nodes, leading to a low accuracy for the search.

F. Effect of Parameters

In this section, we study the effect of the parameters used in Cheetah and Cheetah⁺. Since the results of Cheetah and Cheetah⁺ are similar, we only present the results of Cheetah.

Effect of Adaptive Beam search Threshold γ. We conduct a grid search for the threshold γ used in our adaptive beam search algorithm from 0 to 1 with stride 0.1, and measure the accuracy and response time of the search algorithm under different γ. The results are depicted in Figures 12 and 13. We can see that when γ is greater than 0.4, the accuracy begin to decline rapidly while the response time tends to stabilize. Thus, we choose $\gamma = 0.4$ to balance accuracy and efficiency.

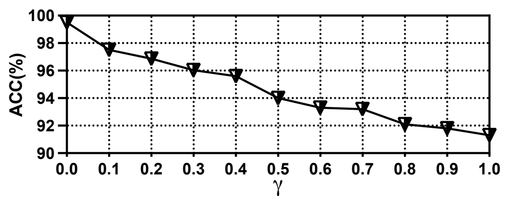


Fig. 12. Effect of threshold γ (accuracy)

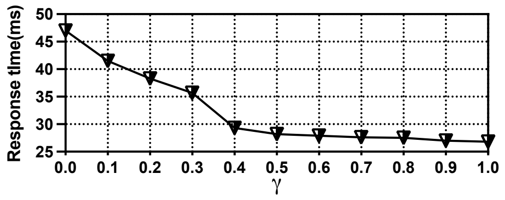


Fig. 13. Effect of threshold γ (response time)

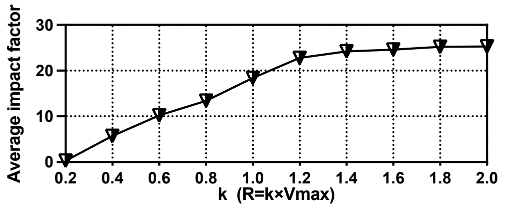


Fig. 14. Effect of radius R (average impact factor)

Effect of Impact Factor Radius R. The radius of the impact factor determines the size of the nearby environment that the autonomous vehicle needs to consider. An overly large radius will cause calculation redundancy; conversely, an overly small radius will ignore important nearby vehicles. And intuitively, the radius is proportional to the maximal speed for the road. Therefore, the radius is formalized as: $R = k \times V_{max}$. We test the parameter k at different V_{max} values, i.e., highway=32m/s, urban=15m/s, 20m/s and downtown=5m/s, so as to calculate the average impact factor under different R. We took the average value of the impact factor under each candidate value of V_{max} to compare the different values of k. As shown in Figure 14, when k equals to 1.2, the increasing trend of the impact factor slows down, which means the impact factor tends to stabilize. Thus, $k = 1.2$ is taken as the default. Correspondingly, we set the radius in the experiments as $R = 38m$ w.r.t. V_{max}=32m/s ($115km/h$).

VII. RELATED WORK

Lane change is one of the most conventional behaviors in autonomous driving. In this study, an autonomous vehicle is supposed to plan a series of lane change maneuvers. Existing lane change planing studies typically consider the driving safety and comfort of the vehicles [5], [6], [34], [35]. For example, [5] presents a novel control algorithm for lane change assistance and autonomous driving on highways based on Scenario Model Predictive Control (SCMPC). The basic idea is to account for the uncertainty in the traffic environment by a small number of future scenarios to perform safe lane change. [34] proposes a classical cellular automata model (called STNS) for planning the behaviors of vehicles, where a set of rules are applied to judgement the future lane change maneuvers. [6] presents a kinematics model for lane change, which can plan the trajectories based on the characteristics of polynomials. Besides, the infinite dynamic circles are applied to detect collision during lane change. [35] proposes a Freeway LaneSelection model (FLS), which will enable transportation professionals to more accurately model lane-changing behaviors on freeways. FLS algorithm consists of choice of target lane and gap acceptance decisions, which aims to output the most accurate lane change maneuver. Our study focuses on the impact of the lane change maneuver of the autonomous vehicle, aiming to alleviate 'phantom' traffic jams.

Urban computing [45] aims to solve the issues caused by human's rapid progress in urbanization, such as traffic signal control [46]–[48], bike lane planning recommendation [49], [50], crime rate inference [51], crowd flow alert [52], [53], and resource rebalancing [54]. This study aims to optimize the lane changing maneuvers of autonomous vehicles, thereby enhancing the overall efficiency and safety of autonomous driving in urban or highway environments.

VIII. Conclusion

In this paper, we propose Cheetah, a prediction-and-decision framework that helps the autonomous vehicle change lanes more wisely. To make our framework applicable to more scenarios, we further propose Cheetah$^+$ that makes the autonomous vehicle adapt for exiting a road and meet the requirement for driving comfort. Extensive experiments based on both real and synthetic datasets also confirm the superiority of our proposed framework over state-of-the-art approaches in terms of both macroscopic and microscopic effectiveness.

Acknowledgment

This work is partially supported by NSFC (No. 61972069, 61836007 and 61832017), Shenzhen Municipal Science and Technology R&D Funding Basic Research Program (JCYJ20210324133607021), and Municipal Government of Quzhou under Grant (No. 2022D037, 2023D044), and Key Laboratory of Data Intelligence and Cognitive Computing, Longhua District, Shenzhen.

References

[1] D. Schrank, B. Eisele, and T. Lomax, "2019 urban mobility report," Texas A&M Transportation Institute, Tech. Rep., 2019.

[2] M. Won, T. Park, and S. H. Son, "Toward mitigating phantom jam using vehicle-to-vehicle communication," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1313–1324, 2017.

[3] L. Wang and B. K. P. Horn, "On the stability analysis of mixed traffic with vehicles under car-following and bilateral control," *IEEE Transactions on Automatic Control*, vol. 65, no. 7, pp. 3076–3083, 2020.

[4] L. Davis, "Effect of adaptive cruise control systems on traffic flow," *Physical Review E*, vol. 69, no. 6, p. 066110, 2004.

[5] G. Cesari, G. Schildbach, A. Carvalho, and F. Borrelli, "Scenario model predictive control for lane change assistance and autonomous driving on highways," *IEEE Intelligent transportation systems magazine*, vol. 9, no. 3, pp. 23–35, 2017.

[6] F. You, R. Zhang, G. Lie, H. Wang, H. Wen, and J. Xu, "Trajectory planning and tracking control for autonomous lane change maneuver based on the cooperative vehicle infrastructure system," *Expert Systems with Applications*, vol. 42, no. 14, pp. 5932–5946, 2015.

[7] Tesla, "Model s owner's manual," 2022. [Online]. Available: https://www.tesla.com/ownersmanual/models/en_us/

[8] X. Li and J.-Q. Sun, "Studies of vehicle lane-changing dynamics and its effect on traffic efficiency, safety and environmental impact," *Physica A: Statistical Mechanics and its Applications*, vol. 467, pp. 41–58, 2017.

[9] SAE-International, "Sae standards news: J3016 automated-driving graphic update," 2019. [Online]. Available: https://www.sae.org/news/2019/01/sae-updates-j3016-automated-driving-graphic

[10] S. Liu, H. Su, Y. Zhao, K. Zeng, and K. Zheng, "Lane change scheduling for autonomous vehicle: A prediction-and-search framework," in *SIGKDD*, 2021, pp. 3343–3353.

[11] M. Zhu, Y. Wang, Z. Pu, J. Hu, X. Wang, and R. Ke, "Safe, efficient, and comfortable velocity control based on reinforcement learning for autonomous driving," *Transportation Research Part C: Emerging Technologies*, vol. 117, p. 102662, 2020.

[12] T. Toledo and D. Zohar, "Modeling duration of lane changes," *Transportation Research Record*, vol. 1999, no. 1, pp. 71–78, 2007.

[13] R. Worrall and A. Bullen, "An empirical analysis of lane changing on multilane highways," *Highway Research Record*, no. 303, 1970.

[14] S. E. Lee, E. C. Olsen, W. W. Wierwille *et al.*, "A comprehensive examination of naturalistic lane-changes," United States. National Highway Traffic Safety Administration, Tech. Rep., 2004.

[15] M. Rickert, K. Nagel, M. Schreckenberg, and A. Latour, "Two lane traffic simulations using cellular automata," *Physica A: Statistical Mechanics and its Applications*, vol. 231, no. 4, pp. 534–550, 1996.

[16] M. M. Pedersen and P. T. Ruhoff, "Entry ramps in the nagel-schreckenberg model," *Physical Review E*, p. 056705, 2002.

[17] A. K. Daoudia and N. Moussa, "Numerical simulations of a three-lane traffic model using cellular automata," *Chinese journal of physics*, vol. 41, no. 6, 2003.

[18] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi, "Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1672–1678.

[19] N. Deo and M. M. Trivedi, "Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1179–1184.

[20] F. Altché and A. de La Fortelle, "An lstm network for highway trajectory prediction," in *ITSC*. IEEE, 2017, pp. 353–359.

[21] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.

[22] Y. Zha, J. Deng, Y. Qiu, K. Zhang, and Y. Wang, "A survey of intelligent driving vehicle trajectory tracking based on vehicle dynamics," *SAE International journal of vehicle dynamics, stability, and NVH*, vol. 7, no. 10-07-02-0014, 2023.

[23] Q. Shi and H. Zhang, "An improved learning-based lstm approach for lane change intention prediction subject to imbalanced data," *Transportation research part C: emerging technologies*, vol. 133, p. 103414, 2021.

[24] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[25] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *arXiv preprint arXiv:1402.1128*, 2014.

[26] C. Raffel and D. P. Ellis, "Feed-forward networks with attention can solve some long-term memory problems," *arXiv preprint arXiv:1512.08756*, 2015.

[27] K. Schofield, "Automotive lane change aid," Apr. 19 2005, uS Patent 6,882,287.

[28] R. Herman, E. W. Montroll, R. B. Potts, and R. W. Rothery, "Traffic dynamics: analysis of stability in car following," *Operations research*, vol. 7, no. 1, pp. 86–106, 1959.

[29] A. Ahmed, F. Outay, S. O. R. Zaidi, M. Adnan, and D. Ngoduy, "Examining queue-jumping phenomenon in heterogeneous traffic stream at signalized intersection using uav-based data," *Personal and Ubiquitous Computing*, pp. 1–16, 2020.

[30] F. O. de França, "A greedy search tree heuristic for symbolic regression," *Information Sciences*, vol. 442, pp. 18–32, 2018.

[31] M. Freitag and Y. Al-Onaizan, "Beam search strategies for neural machine translation," in *Proceedings of the First Workshop on Neural Machine Translation*, Aug. 2017, pp. 56–60.

[32] C. Dong, H. Wang, Y. Li, W. Wang, and Z. Zhang, "Route control strategies for autonomous vehicles exiting to off-ramps," *IEEE TITS*, vol. 21, no. 7, pp. 3104–3116, 2019.

[33] F. Hartwich, M. Beggiato, and J. F. Krems, "Driving comfort, enjoyment and acceptance of automated driving–effects of drivers' age and driving style familiarity," *Ergonomics*, vol. 61, no. 8, pp. 1017–1032, 2018.

[34] D. Chowdhury, D. E. Wolf, and M. Schreckenberg, "Particle hopping models for two-lane traffic with two kinds of vehicles: Effects of lane-changing rules," *Physica A: Statistical Mechanics and its Applications*, vol. 235, no. 3-4, pp. 417–439, 1997.

[35] C. Choudhury, T. Toledo, and M. Ben-Akiva, "Ngsim freeway lane selection model," Federal Highway Administration, Tech. Rep., 12 2004.

[36] I. Jacobson, L. Richards, and A. Kuhlthau, "Models of human comfort in vehicle environments," *Human Factors in Transport Research Edited by DJ Oborne, JA Levis*, vol. 2, 1980.

[37] H. Yeo, A. Skabardonis, J. Halkias, J. Colyar, and V. Alexiadis, "Oversaturated freeway flow algorithm for use in next generation simulation," *Transportation Research Record*, vol. 2088, no. 1, pp. 68–79, 2008.

[38] L. Zhang, S. Cai, Y. Zhang, and M. Zhang, "Comparison of lane changing algorithms between ngsim and corsim," in *2010 IEEE 71st Vehicular Technology Conference*. IEEE, 2010, pp. 1–6.

[39] K. Nagel, D. E. Wolf, P. Wagner, and P. Simon, "Two-lane traffic rules for cellular automata: A systematic approach," *Physical Review E*, vol. 58, no. 2, p. 1425, 1998.

[40] J. Colyar and J. Halkias, "Next generation simulation ngsim us highway 101 dataset," Federal Highway Administration, Tech. Rep., 2007.

[41] J. Halkias and J. Colyar, "Next generation simulation ngsim interstate 80 freeway dataset," Federal Highway Administration, Tech. Rep., 2006.

[42] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 1468–1476.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[44] P. Bokare and A. Maurya, "Acceleration-deceleration behaviour of various vehicle types," *Transportation research procedia*, vol. 25, pp. 4733–4749, 2017.

[45] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: concepts, methodologies, and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 3, pp. 1–55, 2014.

[46] X. Li and J.-Q. Sun, "Signal multiobjective optimization for urban traffic network," *IEEE TITS*, vol. 19, no. 11, pp. 3529–3537, 2018.

[47] C. Chen, H. Wei, N. Xu, G. Zheng, M. Yang, Y. Xiong, K. Xu, and Z. Li, "Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control," in *AAAI*, 2020, pp. 3414–3421.

[48] X. Li and J.-Q. Sun, "Multi-objective optimal predictive control of signals in urban traffic network," *Journal of Intelligent Transportation Systems*, vol. 23, no. 4, pp. 370–388, 2019.

[49] J. Bao, T. He, S. Ruan, Y. Li, and Y. Zheng, "Planning bike lanes based on sharing-bikes' trajectories," in *ACM SIGKDD*, 2017, pp. 1377–1386.

[50] T. He, J. Bao, S. Ruan, R. Li, Y. Li, H. He, and Y. Zheng, "Interactive bike lane planning using sharing bikes' trajectories," *IEEE TKDE*, vol. 32, no. 8, pp. 1529–1542, 2019.

[51] H. Wang, D. Kifer, C. Graif, and Z. Li, "Crime rate inference with big data," in *ACM SIGKDD*, 2016, pp. 635–644.

[52] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *AAAI*, 2017.

[53] C. Shi, X. Han, L. Song, X. Wang, S. Wang, J. Du, and S. Y. Philip, "Deep collaborative filtering with multi-aspect information in heterogeneous networks," *IEEE TKDE*, pp. 1413–1425, 2019.

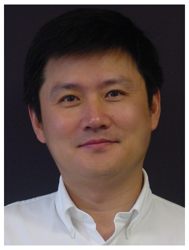
[54] J. Liu, L. Sun, W. Chen, and H. Xiong, "Rebalancing bike sharing systems: A multi-source data smart optimization," in *ACM SIGKDD*, 2016, pp. 1005–1014.

Shuncheng Liu received the Bachelor degree in Computer Science and Technology from Chongqing University of Posts and Telecommunications, in 2019. He is currently a PhD student in University of Electronic Science and Technology of China. His research interests include autonomous driving and spatio-temporal data mining.

Xu Chen recieved the Bachelor degree in Electronics and Electrical Engineering from University of Electronic Science and Technology of China, in 2020. He is now a PhD student in the same university. His research interests include spatio-temporal data mining and AI for database.

Yan Zhao is an Assistant Professor with Aalborg University. She received the PhD Degree in Computer Science from Soochow University in 2020. Her research interests include spatial database and trajectory computing.

Han Su received the BS degree in software engineering from Nanjing University, in 2011, and the PhD degree in computer science from the University of Queensland, in 2015. She is currently an associate professor with University of Electronic Science and Technology of China. Her research interests include autonomous driving and trajectory mining.



Xiaofang Zhou received the bachelor's and master's degrees in computer science from Nanjing University, in 1984 and 1987, respectively, and the PhD degree in computer science from the University of Queensland in 1994. He is the Otto Poon Professor of Engineering and Chair Professor of Computer Science and Engineering at the Hong Kong University of Science and Technology. His research is focused on finding effective and efficient solutions to managing integrating, and analyzing very large amounts of complex data for business and scientific applications. His research interests include spatial and multimedia databases, high performance query processing, web information systems, data mining, and data quality management. He is a fellow of IEEE.

Kai Zheng is a Professor of Computer Science with University of Electronic Science and Technology of China. He received his PhD degree in Computer Science from The University of Queensland in 2012. He has been working in the area of spatial-temporal databases, uncertain databases, social-media analysis, in-memory computing, and blockchain technologies. He has published over 100 papers in prestigious journals and conferences in data management field such as SIGMOD, ICDE, VLDB Journal, ACM Transactions and IEEE Transactions. He is a senior member of IEEE.