2023 IEEE 39th International Conference on Data Engineering (ICDE) | 979-8-3503-2227-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICDE55515.2023.00250

Impact-aware Maneuver Decision with Enhanced Perception for Autonomous Vehicle

Shuncheng Liu[1], Yuyang Xia[1], Xu Chen[1], Jiandong Xie[2], Han Su[1,3,*], Kai Zheng[1,4,*]
[1]School of Computer Science and Engineering, University of Electronic Science and Technology of China, China
[2]Huawei Cloud Database Innovation Lab, China
[3]Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, China
[4]Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, China
{liushuncheng,xiayuyang,xuchen}@std.uestc.edu.cn, xiejiandong@huawei.com, {hansu,zhengkai}@uestc.edu.cn

***Abstract*—Autonomous driving is an emerging technology that has developed rapidly over the last decade. There have been numerous interdisciplinary challenges imposed on the current transportation system by autonomous vehicles. In this paper, we conduct an algorithmic study on the autonomous vehicle decision-making process, which is a fundamental problem in the vehicle automation field and the root cause of most traffic congestion. We propose a perception-and-decision framework, called *HEAD*, which consists of an enHanced pErception module and a mAneuver Decision module. *HEAD* aims to enable the autonomous vehicle to perform safe, efficient, and comfortable maneuvers with minimal impact on other vehicles. In the enhanced perception module, a graph-based state prediction model with a strategy of phantom vehicle construction is proposed to predict the one-step future states for multiple surrounding vehicles in parallel, which deals with sensor limitations such as limited detection range and poor detection accuracy under occlusions. Then in the maneuver decision module, a deep reinforcement learning-based model is designed to learn a policy for the autonomous vehicle to perform maneuvers in continuous action space w.r.t. a parameterized action Markov decision process. A hybrid reward function takes into account aspects of safety, efficiency, comfort, and impact to guide the autonomous vehicle to make optimal maneuver decisions. Extensive experiments offer evidence that *HEAD* can advance the state of the art in terms of both macroscopic and microscopic effectiveness.**

***Index Terms*—Autonomous driving, Perception, Decision**

I. Introduction

Most major cities worldwide experience high levels of traffic congestion due to the rapid development in urbanization and vehicle ownership [1]. Typically, road environments or drivers are to blame for traffic congestion [2]. Environmental variables like road construction, traffic lights, or a reduction in the number of lanes (bottleneck) can give rise to traffic congestion. Additionally, a driver's poor driving behavior (e.g., hard braking and forced lane change) may result in traffic congestion or even accidents. The latter is more frequent due to the differences in drivers' habits [3]. It usually happens when the traffic density is high, thus even a slight fluctuation in the traffic flow can generate a 'domino effect' and lead to serious traffic congestion. To avoid such phenomenon, drivers need to keep good driving behaviors and maintain a safe distance between vehicles [4], which are highly challenging, if not impossible, for human drivers.

With the rapid development of vehicle automation technology, this goal may be achieved in the future when a considerable portion of on-road vehicles are autonomous vehicles. Some dangerous driving behaviors such as speed driving and drowsy driving can be avoided by gradually replacing human control with autonomous decision-making algorithms [5]. Traditional methods have demonstrated that autonomous vehicles can maintain a constant distance from surrounding vehicles with the aid of adaptive cruise control and lane-changing models [6]–[8]. However, these methods involve a set of rule-matching algorithms and require expert experience and manual tuning, leading to poor generalizability with the increasing complexity of autonomous driving scenarios. Considering the mechanism that a driver perceives the surrounding traffic and makes a maneuver decision (lane change behavior and/or velocity change behavior), it fits well within the realm of reinforcement learning [9]. Due to the flexible reward designing and superior optimization effect, there have been plenty of works utilizing reinforcement learning-based methods to accomplish vehicle maneuver decisions in the scenario of autonomous driving [10]–[13]. These reinforcement learning-based approaches mainly optimize the driving safety, efficiency, and comfort of autonomous vehicles, leaving the impact on other surrounding vehicles and eventually traffic conditions largely uninvestigated. Evidently, if autonomous vehicles make maneuver decisions simply based on their states without taking the driving conditions of surrounding vehicles into account, they may cause more serious traffic congestion or even accidents. Recently, a prediction-and-search framework [14] is proposed to make discrete maneuver decisions, which considers three impact situations of an autonomous vehicle on its surrounding vehicles, including queuing, crossing, and jumping the queue. However, it ideally discretizes the velocity change behavior as speed-up, speed-down, and maintain speed, which lacks effectiveness in continuous action space. Further, it still relies on hand-crafted rules for determining different impact situations, which cannot deal with the impact of continuous velocity change behavior [15]. Therefore, existing decision-making algorithms for autonomous driving

* Corresponding authors: Kai Zheng and Han Su

2375-026X/23/$31.00 ©2023 IEEE
DOI 10.1109/ICDE55515.2023.00250

cannot effectively reduce the impact on surrounding vehicles.

To sum up, an ideal decision framework for the autonomous vehicle can perform safe and comfortable maneuvers with high driving efficiency and minimal impact on surrounding vehicles. In particular, reducing the impact plays a vital role in addressing poor driving behaviors and reducing traffic congestion or accidents. Intuitively, one can embed a trajectory prediction model [14], [16] in perception modules (with onboard sensors), which not only capture the current state of surrounding vehicles, but also proactively anticipate their future behaviors, and then utilize a deep reinforcement learning-based model to make maneuver decisions [9]. However, the intuition will face two main challenges: (1) The states of surrounding vehicles are not always observable due to sensor limitations like detection range and occlusion, making the trajectory prediction models less effective. (2) Reinforcement learning-based models struggle to balance the factors of safety, efficiency, comfort, and impact for complex vehicle maneuvers, and it is also challenging to measure the impact factor. In this work, we aim to address the above challenges and enable the autonomous vehicle to perform safe and comfortable maneuvers while maximizing its average velocity and minimizing its impact on surrounding vehicles.

To this end, we proposed a novel perception-and-decision framework, called *HEAD*, which consists of an enHanced pErception module and a mAneuver Decision module. In the enhanced perception module, we propose a state prediction model to predict the one-step future states for multiple surrounding vehicles in parallel. To deal with the incomplete historical states caused by sensor limitations, it first constructs phantom vehicles based on observable surrounding vehicles and organizes their relationships using spatial-temporal graph, and then utilizes a graph attention mechanism with an LSTM to enable vehicle interactions and parallel prediction. For the maneuver decision module, it first receives the future states of surrounding vehicles and formulates the maneuver decision task as a Parameterized Action Markov Decision Process (PAMDP) with discrete lane change behaviors and continuous velocity change behavior, and then uses a deep reinforcement learning-based model and a properly designed reward function to solve the PAMDP, which learn an optimized policy for the autonomous vehicle to achieve our objective. In summary, we make the following contributions:

- We develop a perception-and-decision framework that enables the autonomous vehicle to perform safe, efficient, and comfortable maneuvers with minimal impact on other vehicles.
- We propose a graph-based state prediction model with a strategy of phantom vehicle construction to solve sensor limitations and support high-accuracy prediction in parallel.
- We propose a deep reinforcement learning-based model and a hybrid reward function to make maneuver decisions in continuous action space that follows a parameterized action Markov decision process.
- We conduct extensive experiments to evaluate *HEAD* on real and simulated data, verifying the effectiveness on both macroscopic and microcosmic metrics.

II. Overview

A. Preliminary Concepts

Environment. We consider an interactive environment where there are one autonomous vehicle A and a set of conventional vehicles $\mathbb{C}$ traveling 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 states (i.e., locations and velocities) of surrounding conventional vehicles through its sensors, and perform a maneuver at each time instant t within a target time duration $\mathbb{T}$ of interest.

Lane. A lane is part of the road used to guide vehicles in the same direction. Herein, all the lanes are numbered incrementally from the leftmost side to the rightmost side, i.e., $l_1, l_2, \ldots, l_\kappa$, where l_1 and l_κ indicate the leftmost lane and rightmost lane, respectively.

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., $\mathbb{T} = \{1, 2, \ldots, t, \ldots\}$. We denote Δt as the time interval between two consecutive time steps, which serves as the minimum frequency for the autonomous vehicle to perform maneuvers. Following the settings used in the previous work [14], [17], the time granularity in this work is set to 0.5 seconds (i.e., $\Delta t = 0.5s$).

Location. $(C_i^t.lat, C_i^t.lon)$ and $(A^t.lat, A^t.lon)$ indicate the locations of C_i and A, respectively, at time step t, where lat denotes the lateral lane number and lon refers to the longitudinal location of a vehicle traveled from the origin. $d_{lon}(C_i^t, A^t)$ denotes the relative longitudinal distance between C_i and A at time step t, which can be calculated as follows:

$$d_{lon}(C_i^t, A^t) = C_i^t.lon - A^t.lon \tag{1}$$

In addition, the $d_{lat}(C_i^t, A^t)$ denotes the relative lateral distance between C_i and A at time step t, which can be calculated as follows:

$$d_{lat}(C_i^t, A^t) = (C_i^t.lat - A^t.lat) * wid_l \tag{2}$$

where wid_l is the width of a lane. An advantage of using this type of lane-aware location is to allow us to focus on the lane change behavior itself without worrying about the lateral location of the vehicle.

Velocity. $C_i^t.v$ and $A^t.v$ indicate the longitudinal velocities of C_i and A, respectively, at time step t. $v(C_i^t, A^t)$ denotes the relative longitudinal velocity between C_i and A at time step t, which can be calculated as follows:

$$v(C_i^t, A^t) = C_i^t.v - A^t.v \tag{3}$$

Benefiting from the discrete time step and the lane-aware location, the lateral motion between two consecutive time steps is assumed to be the uniform motion [14], [18], so we focus on the longitudinal velocity in this work. In the rest of the paper, we use velocity and longitudinal velocity interchangeably when no ambiguity is caused.

Maneuver. A maneuver is a pair of a *lateral lane change behavior* and a *longitudinal velocity change behavior* simultaneously performed by a vehicle [19]. $(A^t.b, A^t.a)$ represents

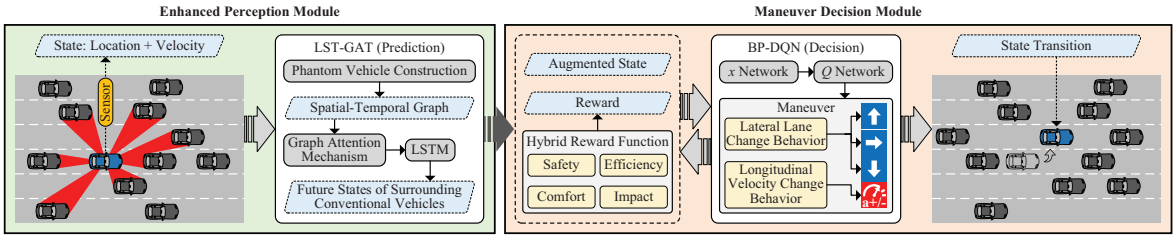

Fig. 1. *HEAD* Framework Overview

the maneuver of A at time step t, where b is one of the three lateral lane change behaviors: *change lane to left* (ll), *change lane to right* (lr), and *lane keep* (lk) (i.e., $b \in \{ll, lr, lk\}$), and a refers to the longitudinal acceleration. Since ll and lr are assumed to be the uniform motions, the lateral acceleration can be ignored.

Restrictions. We pose some traffic restrictions on all vehicles:
(**1**) Speed limit. All lanes are subject to two speed limit, i.e., v_{min} and v_{max}.
(**2**) Lane change restriction. A vehicle can only change to an adjacent lane at each time step.
(**3**) Velocity change restriction. The longitudinal acceleration of a vehicle is bounded between $-a'$ and a'.

Objective. In this work, our objective is that *the autonomous vehicle can perform safe and comfortable maneuvers while maximizing its average velocity and minimizing its impact on surrounding conventional vehicles*.

B. Framework Overview

Figure 1 shows the architecture of our framework *HEAD*, which consists of two components: *enhanced perception module* and *maneuver decision module*. Firstly, the enhanced perception module obtains the states of surrounding conventional vehicles through the onboard sensor, a graph-based state prediction model (LST-GAT) constructs phantom vehicles based on observable surrounding vehicles and organizes their relationships using spatial-temporal graph structure, and then utilizes a graph attention mechanism with an LSTM to predict future states of surrounding conventional vehicles in parallel. Secondly, the future states are fed into the maneuver decision module as augmented states, and then a deep reinforcement learning-based model (BP-DQN) and a hybrid reward function are proposed to learn a policy for the autonomous vehicle to perform maneuvers with discrete lateral lane change behaviors and continuous longitudinal velocity change behavior.

III. Enhanced Perception

To reduce the impact of the autonomous vehicle on surrounding conventional vehicles, it is necessary for an automotive perception module to predict the future trajectories of surrounding conventional vehicles. However, simply embedding a trajectory prediction model in a perception module will face poor applicability, accuracy, and efficiency in the scenario of autonomous driving. Thus, we propose an enhanced perception module that can make use of the limited historical states of the surrounding conventional vehicles to predict their future states with high accuracy and efficiency. In this section, we first analyze the limitations of existing perception modules and trajectory prediction models, and then we propose a graph-based deep learning model with a strategy of phantom vehicle construction for achieving our state prediction task.

A. Limitations of Existing Methods

Traditional Perception Modules. The traditional perception modules of the autonomous vehicle perceive the interactions with the surrounding conventional vehicles through sensors [12], [20], [21], e.g., cameras, LiDAR, Ultrasonic, etc. Specifically, they acquire the real-time states (i.e., locations and velocities) of the surrounding conventional vehicles [22], and feed them into the decision modules [14] to make maneuver decisions [23]. However, they struggle to achieve our objective due to the following reasons: (1) They discard more historical states of conventional vehicles that imply their driving preferences and moving tendencies. (2) They ignore the future states of conventional vehicles that reflect their possible changes in driving conditions. Apparently, without taking these enriched states into account, the autonomous vehicle cannot reduce the impact on the surrounding conventional vehicles, and may cause more traffic congestion or even accidents.

Trajectory Prediction Models. In order to remedy the above problems, the most straightforward solution is to utilize the sequence of historical states to predict the sequence of future states (namely future trajectory) for each surrounding conventional vehicle of the autonomous vehicle, i.e., embedding the trajectory prediction models to the traditional perception modules [14]. Recently, with a success achieved in applying RNN [24] to model non-linear temporal dependencies in sequence learning tasks, there have been plenty of works [25]–[28] utilizing RNN based encoder-decoder architectures to predict the future trajectories of vehicles. However, directly using these learning-based trajectory prediction models usually leads to unsatisfactory results due to the following reasons: (1) Their applicability is poor, since they assume that the states of all surrounding conventional vehicles are always observable, which is not a practical assumption in autonomous driving applications. A more realistic approach should always consider sensor limitations like detection range and occlusion. (2) Their accuracy of the predicted future trajectories decreases over time, and only the first or first few predicted states are reliable. (3) Their inference processes are time-consuming, since they do not support parallel prediction and require separate calculations for each predicted vehicle. Therefore, directly using the trajectory prediction models to enhance

the traditional perception modules will not only reduce the efficiency of perception, but also introduce additional errors for downstream decision modules.

Opportunities. Accordingly, there are three challenges in implementing an ideal perception module: 1) how to deal with the incomplete historical states that follow the sensor limitations; 2) how to ensure the high-accuracy trajectory prediction; and 3) how to speed up the efficiency of trajectory prediction. Next, we will analyze the opportunities that can remedy them.

(**1**) Sensor limitations. Most automotive sensors have limited observability. For example, a LiDAR detects objects at a distance of up to 90m–250m [29], [30], and it cannot accurately detect objects through obstacles due to the weak reflection [31], [32]. Other sensors like cameras and RADAR also have limited detection ranges and poor detection accuracy under occlusions [22], [33]. Thus, the availability of most trajectory prediction models will be affected by incomplete input states as they assume that all surrounding vehicles are always observable. If we simply ignore the unobservable vehicles and only use the valid input states, the trajectory prediction results will be unconstrained, causing inaccurate predictions. More importantly, the autonomous vehicle may encounter blind-spot accidents due to inaccurate predictions and unobservable vehicles [34]. Intuitively, we can construct a '*phantom vehicle*' for every blind spot caused by the sensor limitations. Although these vehicles are virtual, they can be preset with suitable locations and velocities, so that a perception module can make trajectory predictions more carefully, and the autonomous vehicle can make safer maneuver decisions.

(**2**) Accuracy. For existing trajectory prediction models, the accuracy of the predicted future trajectories decreases over time, which is caused by two facts. From the physical view, the nearer the predicted future state, the more relevant it is to the historical states [16]. From the modeling view, the sequential decoding schema will accumulate errors over time [35]. The most straightforward solution is to predict the first future state for each surrounding conventional vehicle, which reduces multi-step trajectory prediction to one-step state prediction. This idea is motivated by two aspects. Firstly, one-step prediction not only retains the state with the highest accuracy, but also reduces the structural complexity of a model. Secondly, since the autonomous vehicle has to perceive surroundings and make maneuver decisions at each time step [14], [15], the one-step state prediction is enough for real-time maneuver decisions while the long-term future states are superfluous.

(**3**) Efficiency. Existing trajectory prediction models cannot perform fast surroundings perception for the autonomous vehicle. Specifically, they mainly focus on the trajectory prediction task of one target vehicle [25], [27], [28], [36]. Once the number of surrounding vehicles increases, they need to separately predict the future states of each surrounding vehicle to conduct surroundings perception, leading to poor computational efficiency. This limitation can be solved by designing a lightweight model that can perform parallel state prediction for multiple surrounding vehicles. The difficulty of parallel prediction is how to decode all prediction results quickly and accurately. One can use multiple decoders to output the results separately or use a single decoder to output the results simultaneously [37], [38]. However, these tricks are difficult to balance efficiency and accuracy due to the sequential decoding of multi-step prediction. In our context, we only need to model the interaction of surrounding vehicles to complete parallel state prediction without redundant decoder computations, benefiting from the idea of one-step prediction with high accuracy.

B. State Prediction for Surrounding Conventional Vehicles

According to the above analysis, we need an enhanced perception module that can 1) deal with incomplete historical states by constructing reasonable phantom vehicles; and 2) utilize the processed historical states to predict the one-step future states for multiple surrounding vehicles in parallel. Thus, we propose a graph-based state prediction model, called LST-GAT (Local Spatial-Temporal Graph ATtention), as the core unit of our enhanced perception module. The main idea is to first construct phantom vehicles based on observable surrounding vehicles and organize their relationships using spatial-temporal graph structure, and then utilize a graph attention mechanism with an LSTM to enable vehicle interactions and parallel prediction. We note that our prediction task faces local spatial states over local time periods caused by sensor limitations, unlike other global spatial-temporal tasks [27], [39]–[41] where spatial states are globally known all the time. In order to adapt to the local spatial-temporal prediction task, we first design a strategy of phantom vehicle construction to deal with incomplete historical states, which improves the applicability of our prediction model. The historical state of the autonomous vehicle, the surrounding vehicles, and the constructed phantom vehicles will be organized as a spatial-temporal graph, which can best preserve the relationships between vehicles during a historical period [42], [43]. Then a graph attention mechanism [44] with an LSTM [45] aggregates the nodes of the spatial-temporal graph with different importance scores and accomplishes parallel state predictions for surrounding vehicles, which accurately and efficiently predicts their one-step future states via a lightweight network structure. Next, we will formally define our state prediction task and give a detailed description of LST-GAT model.

State Prediction Task. Considering the interactive environment introduced in Section II. Our state prediction task is to predict the one-step future states of a set of target conventional vehicles around the autonomous vehicle, conditioning on the historical states of the autonomous vehicle and all surrounding conventional vehicles observed by the onboard sensor.

The problem can be defined as follows: at time step t, given the states (i.e., location and velocity) of the autonomous vehicle A and all observed conventional vehicles $\mathbb{C}_{obs}$ within the historical period of $\{t-z+1, t-z+2, \ldots, t\}$, we predict the states of a set of target conventional vehicles $\mathbb{C}_{tar}$ ($\mathbb{C}_{tar} \subseteq \mathbb{C}_{obs}$) at time step $t+1$. z refers to the number of

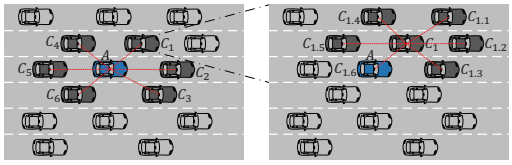

Fig. 2. Example of Target Conventional Vehicles

historical time steps, and $\mathbb{C}_{obs}$ are observed by the sensor of A within the detection range $\mathcal{R}$.

Based on the definition, at time step t, the inputs are 1) $(C_i^\tau.lat, C_i^\tau.lon), C_i^\tau.v$ of each conventional vehicle $C_i \in \mathbb{C}_{obs}$; and 2) $(A^\tau.lat, A^\tau.lon), A^\tau.v$ of the autonomous vehicle A at each time step $\tau \in \{t-z+1, t-z+2, \ldots, t\}$. The outputs are $(C_i^{t+1}.lat, C_i^{t+1}.lon), C_i^{t+1}.v$ of each conventional vehicle $C_i \in \mathbb{C}_{tar}$ at time step $t+1$.

Phantom Vehicle Construction. The phantom vehicle construction needs to complete three steps. The first step is to select target conventional vehicles from all observed conventional vehicles and check if the target conventional vehicles and their surrounding vehicles are missing. The second step is to construct some phantom vehicles to fill in the missing vehicles based on a properly designed strategy. The final step is to form a spatial-temporal graph using the historical states of all considered vehicles.

(**1**) Step 1. We select target conventional vehicles $\mathbb{C}_{tar}$ from all observed conventional vehicles $\mathbb{C}_{obs}$ that have the most effect on the autonomous vehicle A, and we check if $\mathbb{C}_{tar}$ and their surrounding vehicles are missing due to the sensor limitations. To be specific, we select six target conventional vehicles around the autonomous vehicle as shown on the left side in Figure 2, which covers six key areas centered on A, i.e., 1) front left, 2) front, 3) front right, 4) rear left, 5) rear, and 6) rear right areas. This strategy is widely used [46]–[48] since the six vehicles have the most effect on the decision of the center vehicle. Further, since we need to predict the future states of the target conventional vehicles, the six vehicles around each of them also have the most effect on their future decisions [14]. Thus, we select six surrounding vehicles for each target conventional vehicle as shown on the right side in Figure 2. *Ideally*, $\mathbb{C}_{tar}$ contains 6 vehicles ($\mathbb{C}_{tar} = \{C_1, C_2, \ldots, C_6\}$) and each of them is surrounded by 6 vehicles ($\mathbb{C}_{i.sur} = \{C_{i.1}, C_{i.2}, \ldots, C_{i.6}\}, i \in \{1, 2, \ldots, 6\}$)[1]. However, some of them are always missing due to the sensor limitations. In addition, if a vehicle is in the leftmost lane or rightmost lane, its left or right neighbors will be missing inherently. In step 1, we select $\mathbb{C}_{tar}$ and $\mathbb{C}_{i.sur}$ ($\forall i \in \{1, 2, \ldots, 6\}$) from $\mathbb{C}_{obs}$ and A. If there is a missing vehicle, step 2 starts, otherwise, step 3 starts.

(**2**) Step 2. Once vehicles are found missing, we construct phantom vehicles to fill them based on a strategy. Before introducing the strategy, we identify three missing cases: 1) *range missing*, which is a sensor limitation caused by the limited detection range; 2) *occlusion missing*, which is a sensor limitation caused by the poor detection accuracy under occlusions; and 3) *inherent missing*, which is sensor-independent that a center vehicle is in the leftmost or rightmost lane missing one-side neighbors. As shown in Figure 3, we first deal with missing target conventional vehicles, and then deal with missing vehicles around the target conventional vehicles. Next, we will detail their constructions.

[1]The autonomous vehicle A shares its states with $C_{1.6}$, $C_{2.5}$, $C_{3.4}$, $C_{4.3}$, $C_{5.2}$, and $C_{6.1}$, since each $C_i \in \mathbb{C}_{tar}$ is also surrounded by A.

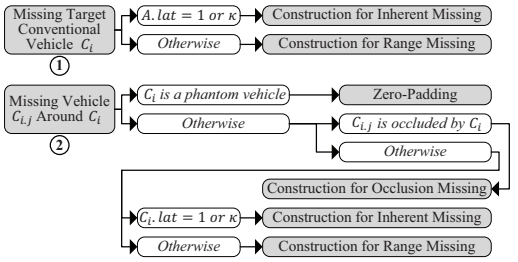

Fig. 3. Strategy of Phantom Vehicle Construction

Firstly, for any missing target conventional vehicle $C_i \in \mathbb{C}_{tar}$, we judge whether it is the range missing or the inherent missing. The two cases can be distinguished by the lateral lane number of the autonomous vehicle. The inherent missing exists when $A.lat$ is 1 (leftmost lane) or κ (rightmost lane), otherwise, the missing target conventional vehicle is the range missing. If it is the range missing, the historical states of C_i will be preset as follows:

$$(C_i^\tau.lat, C_i^\tau.lon), C_i^\tau.v = \begin{cases} (A^\tau.lat - 1, A^\tau.lon + \mathcal{R}), A^\tau.v & A^\tau.lat \neq 1 \text{ and } i = 1 \\ (A^\tau.lat, A^\tau.lon + \mathcal{R}), A^\tau.v & i = 2 \\ (A^\tau.lat + 1, A^\tau.lon + \mathcal{R}), A^\tau.v & A^\tau.lat \neq \kappa \text{ and } i = 3 \\ (A^\tau.lat - 1, A^\tau.lon - \mathcal{R}), A^\tau.v & A^\tau.lat \neq 1 \text{ and } i = 4 \\ (A^\tau.lat, A^\tau.lon - \mathcal{R}), A^\tau.v & i = 5 \\ (A^\tau.lat + 1, A^\tau.lon - \mathcal{R}), A^\tau.v & A^\tau.lat \neq \kappa \text{ and } i = 6 \end{cases} \quad (4)$$

where $\tau \in \{t-z+1, t-z+2, \ldots, t\}$ and $\mathcal{R}$ is the radius of the sensor detection range. The constructed phantom vehicle for the range missing is placed as far as the sensor can detect to make up for the limited detection range. If C_i is the inherent missing, the historical states will be preset as follows:

$$(C_i^\tau.lat, C_i^\tau.lon), C_i^\tau.v = \begin{cases} (0, A^\tau.lon), A^\tau.v & A^\tau.lat = 1 \text{ and } i \in \{1, 4\} \\ (\kappa + 1, A^\tau.lon), A^\tau.v & A^\tau.lat = \kappa \text{ and } i \in \{3, 6\} \end{cases} \quad (5)$$

The constructed phantom vehicle for the inherent missing is placed outside the leftmost and rightmost lanes as a moving road boundary.

Then for any missing vehicle $C_{i.j}$ around C_i ($C_{i.j} \in \mathbb{C}_{i.sur}$), we first judge whether C_i is a constructed phantom vehicle or not. If C_i is a phantom vehicle, the historical states of its surrounding vehicles $\mathbb{C}_{i.sur}$ are filled with zero states (i.e., zero-padding [49]), since they do not need to be constructed based on an uncertain vehicle. If C_i is an observed vehicle, we then determine whether $C_{i.j}$ is the occlusion missing or not. We prioritize the occlusion missing as it is most likely to cause blind-spot accidents [34] compared to the range and inherent missing. Once we find that $C_{i.j}$ may be occluded

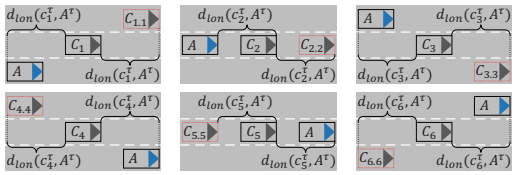

Fig. 4. Examples of Occlusion Missing

by C_i, we construct a phantom vehicle in the occluded area, and the historical states will be preset as follows:

$$(C_{i.j}^{\tau}.lat, C_{i.j}^{\tau}.lon), C_{i.j}^{\tau}.v := \begin{cases} (C_i^{\tau}.lat - 1, C_i^{\tau}.lon + d_{lon}(C_i^{\tau}, A^{\tau})), C_i^{\tau}.v & (i,j) \in \{(1,1),(4,4)\} \\ (C_i^{\tau}.lat, C_i^{\tau}.lon + d_{lon}(C_i^{\tau}, A^{\tau})), C_i^{\tau}.v & (i,j) \in \{(2,2),(5,5)\} \\ (C_i^{\tau}.lat + 1, C_i^{\tau}.lon + d_{lon}(C_i^{\tau}, A^{\tau})), C_i^{\tau}.v & (i,j) \in \{(3,3),(6,6)\} \end{cases} \tag{6}$$

where $d_{lon}(C_i^{\tau}, A^{\tau})$ is the relative longitudinal distance between C_i and A at time step τ defined by Equation (1). The constructed phantom vehicle for the occlusion missing is placed in an occluded position, obeying the basic geometry as shown in Figure 4. If $C_{i.j}$ is not occluded by C_i, it is either the range missing or the inherent missing. Therefore, the phantom vehicle construction for $C_{i.j}$ is similar to that of the missing target conventional vehicle C_i. The historical states of $C_{i.j}$ are preset as Equation (4) and Equation (5) under the range missing and the inherent missing, respectively, by replacing A^{τ} with C_i^{τ}.

(3) Step 3. So far, $\mathbb{C}_{tar}$ and $\mathbb{C}_{i.sur}$ ($\forall i \in \{1,2,\ldots,6\}$) are available, we organize them using spatial-temporal graph structure. In order to better reflect the relationships between the autonomous vehicle and a conventional vehicle, we first replace the historical states of conventional vehicles ($\mathbb{C}_{tar}$ and $\mathbb{C}_{i.sur}$) with the states relative to the autonomous vehicle A, i.e., $(d_{lat}(C^{\tau}, A^{\tau}), d_{lon}(C^{\tau}, A^{\tau})), v(C^{\tau}, A^{\tau})$, which are defined in Section II. The autonomous vehicle still maintains its raw historical state of $(A^{\tau}.lat, A^{\tau}.lon), A^{\tau}.v$ to indicate the reference values. Hereafter, we formulate the historical state vector $h_{C_i^{\tau}} \in \mathbb{R}^4$ of each $C_i \in \mathbb{C}_{tar}$ at each time step $\tau \in \{t-z+1, t-z+2, \ldots, t\}$ as follows:

$$h_{C_i^{\tau}} = [d_{lat}(C_i^{\tau}, A^{\tau}), d_{lon}(C_i^{\tau}, A^{\tau}), v(C_i^{\tau}, A^{\tau}), IF_{C_i}] \tag{7}$$

where IF_{C_i} is a binary code used to indicate if C_i is a phantom vehicle or not. The historical state vector $h_{C_{i.j}^{\tau}} \in \mathbb{R}^4$ of each $C_{i.j} \in \mathbb{C}_{i.sur}$ at time step $\tau \in \{t-z+1, t-z+2, \ldots, t\}$ is formulated as follows:

$$h_{C_{i.j}^{\tau}} = \begin{cases} [A^{\tau}.lat, A^{\tau}.lon, A^{\tau}.v, 0] & (i,j) \in \{(1,6),(2,5),(3,4),(4,3),(5,2),(6,1)\} \\ [d_{lat}(C_{i.j}^{\tau}, A^{\tau}), d_{lon}(C_{i.j}^{\tau}, A^{\tau}), v(C_{i.j}^{\tau}, A^{\tau}), IF_{C_{i.j}}] & otherwise \end{cases} \tag{8}$$

where the first row notes that $C_{1.6}$, $C_{2.5}$, $C_{3.4}$, $C_{4.3}$, $C_{5.2}$, and $C_{6.1}$ use the raw historical states of A (i.e., $h_{A^{\tau}}$), since each $C_i \in \mathbb{C}_{tar}$ is also surrounded by the autonomous vehicle A, and 0 indicates $IF_A = 0$.

Next, considering $\mathbb{C}_{tar}$ and $\mathbb{C}_{i.sur}$ as nodes and their relationships as edges, at time step t, we can construct a spatial-temporal graph as follows:

$$G(t) = [g(t-z+1), g(t-z+2), \ldots, g(t)] \tag{9}$$

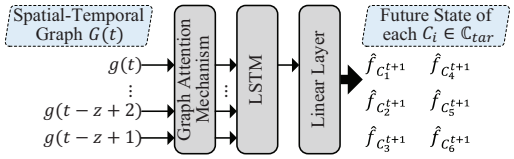

Fig. 5. Network Structure of LST-GAT

where $g(\tau) = (\mathcal{V}(\tau), \mathcal{E})$, $\tau \in \{t-z+1, t-z+2, \ldots, t\}$. $\mathcal{V}(\tau)$ is the node set and each node contains the state vector of a vehicle at time step τ. $\mathcal{E}$ is the time-independent edge set since all considered vehicles are available and their relationships are fixed. The spatial-temporal graph $G(t)$ consists of the spatial graph $g(\tau)$ propagating along the time step τ. To illustrate, ① adding vehicles including $\mathbb{C}_{tar}$, $\mathbb{C}_{i.sur}$ to the node set. So there are 42 ($6 + 6 \times 6$) nodes in the beginning. ② Adding directed edges from C_i to each $C_{i.j} \in \mathbb{C}_{i.sur}$ $\forall i \in \{1,2,\ldots,6\}$. ③ Adding directed edges from C_i to C_i (self-loops). Now a spatial graph is completed. We repeat ①–③ z times (i.e., from $t-z+1$ to t) to form the spatial-temporal graph $G(t)$, which will be fed to the network.

Network Structure. The network structure of LST-GAT consists of a graph attention mechanism and an LSTM as shown in Figure 5. Attention mechanisms have been verified to be effective in various graph-based tasks [44], [50]–[52]. Among them, the graph attention mechanism [44] especially inspires us since it shows an effective approach to capture the spatial relationships between nodes without requiring costly matrix operations. Thus, we first use a graph attention mechanism to aggregate some nodes of each spatial graph in the spatial-temporal graph with different importance scores, which updates the nodes of target conventional vehicles by interacting with their surrounding vehicles. Then we input the updated nodes of target conventional vehicles at each historical time step to an LSTM, which effectively captures their latent temporal dependencies [26], [45] and outputs their future states in parallel. We will introduce our graph attention mechanism and LSTM separately.

Firstly, we apply a sharing attention mechanism to deal with each spatial graph in the spatial-temporal graph. It receives $G(t)$ and produces the updated historical state vectors of all target conventional vehicles. To be specific, we update all target conventional vehicles by computing importance scores of each surrounding vehicle of a target conventional vehicle (including itself) and aggregate their historical states using the different importance scores. Following the standard procedure [43], [44], for each $g(\tau)$ ($\forall \tau \in \{t-z+1, t-z+2, \ldots, t\}$), the importance scores $\alpha_{C_{ix}^{\tau}}$ of each $C_{ix} \in \mathbb{C}_{i.sur} \cup \{C_i\}$ ($\forall i \in \{1,2,\ldots,6\}$) is computed as follows:

$$\alpha_{C_{ix}^{\tau}} = \frac{exp(LeakyReLU(\phi_2[\phi_1 h_{C_i^{\tau}} \parallel \phi_1 h_{C_{ix}^{\tau}}]))}{\sum_{C_{ix} \in \mathbb{C}_{i.sur} \cup \{C_i\}} exp(LeakyReLU(\phi_2[\phi_1 h_{C_i^{\tau}} \parallel \phi_1 h_{C_{ix}^{\tau}}]))} \tag{10}$$

where $\phi_1 \in \mathbb{R}^{D_{\phi_1} \times 4}$ is the linear transformation for historical states (D_{ϕ_1} denotes the dimensions), $\parallel$ is the concatenation operation, and $\phi_2 \in \mathbb{R}^{2D_{\phi_1}}$ is the linear transformation for computing an attentional coefficient. Then we update the

historical states of each target conventional vehicle $C_i \in \mathbb{C}_{tar}$ as follows:

$$h'_{C_i^\tau} = \sum_{C_{ix} \in \mathbb{C}_{i.sur} \cup \{C_i\}} \alpha_{C_{ix}^\tau} * \phi_3 h_{C_{ix}^\tau} \tag{11}$$

where $*$ represents the element-wise production, $\phi_3 \in \mathbb{R}^{D_{\phi_3} \times 4}$ is the linear transformation for historical states (4 refers to the dimensions of $h_{C_{ix}^\tau}$).

Secondly, we use an LSTM [45] to capture the latent temporal dependencies of the updated historical states and output the future state of each target conventional vehicle in parallel. Specifically, the updated historical state $h'_{C_i^\tau}$ of each target conventional vehicle $C_i \in \mathbb{C}_{tar}$ at each time step $\tau \in \{t-z+1, t-z+2, \ldots, t\}$ is fed into an LSTM, which outputs the hidden state vector at each time step as follows:

$$h''_{C_i^\tau} = LSTM(h'_{C_i^\tau}, h''_{C_i^{\tau-1}}; W_l) \tag{12}$$

where W_l denotes the learnable parameters of LSTM and $h''_{C_i^\tau} \in \mathbb{R}^{D_l}$ (D_l is the dimensions of LSTM hidden state vectors), and $h''_{C_i^{\tau-1}}$ defaults to zeros when τ is $t-z+1$. Then the predicted future state $\hat{f}_{C_i^{t+1}}$ of each target conventional vehicle $C_i \in \mathbb{C}_{tar}$ at time step $t+1$ is computed as follows:

$$\hat{f}_{C_i^{t+1}} = \phi_4 h''_{C_i^t} + b_4 \tag{13}$$

where $\phi_4 \in \mathbb{R}^{3 \times D_l}$ is the linear transformation for the final hidden state vector $h''_{C_i^t}$ and b_4 is the bias of ϕ_4. $\hat{f}_{C_i^{t+1}}$ can be expanded out as $[\hat{d}_{lat}(C_i^{t+1}, A^t), \hat{d}_{lon}(C_i^{t+1}, A^t), \hat{v}(C_i^{t+1}, A^t)]$, which refers to the predicted relative states of a target conventional vehicle at time step $t+1$ relative to the autonomous vehicle at time step t. We note that Equation (12) and Equation (13) support the parallel computation of all target conventional vehicles [53] with batched sequences.

Prediction Objective. Evidently, the prediction of $\hat{f}_{C_i^{t+1}}$ ($\forall C_i \in \mathbb{C}_{tar}$) is a regression task. In the duration of interest $\mathbb{T}$, our network needs to minimize the loss function as follows:

$$\mathcal{L}_1 = \frac{1}{|\mathbb{T}|} \sum_{t+1 \in \mathbb{T}} \sum_{C_i \in \mathbb{C}_{tar}} (f_{C_i^{t+1}} - \hat{f}_{C_i^{t+1}})^2 \tag{14}$$

where $|\mathbb{T}|$ denotes the length of $\mathbb{T}$, and $f_{C_i^{t+1}}$ denotes the ground truth of $[d_{lat}(C_i^{t+1}, A^t), d_{lon}(C_i^{t+1}, A^t), v(C_i^{t+1}, A^t)]$. In particular, if C_i is a constructed phantom vehicle, we will set the ground truth as $f_{C_i^{t+1}} = \hat{f}_{C_i^{t+1}}$ to mask its loss.

IV. Maneuver Decision

Borrowing strengths from the enhanced perception module, the future states of target conventional vehicles are fed into the decision module as augmented states. This allows the decision model to make predictive decisions in dynamic environments. In this section, we first formulate our maneuver decision task as a Parameterized Action Markov Decision Process (PAMDP) with a discrete-continuous hybrid action space. Then we use a deep reinforcement learning-based model and a hybrid reward function to solve this PAMDP, which learns a policy for the autonomous vehicle to perform optimal maneuvers.

A. Parameterized Action Markov Decision Process

According to the definition in Section II, $(A^t.b, A^t.a)$ is the maneuver of A at time step t, where b is one of the three lateral lane change behaviors, i.e., $b \in \{ll, lr, lk\}$), and a refers to the longitudinal acceleration. In our maneuver decision task, the lateral lane change behavior is discrete and the longitudinal acceleration is continuous. We formulate the maneuver decision of the autonomous vehicle as a Parameterized Action Markov Decision Process (PAMDP) [54] with a discrete-continuous hybrid action space, denoted by $\mathcal{M} = (\mathcal{S}_+, \mathcal{A}, \mathcal{T}, r, \gamma)$, where $\mathcal{S}_+$ is a set of augmented states, $\mathcal{A}$ is a parameterized action space, $\mathcal{T}$ is a state transition probability distribution, r is a reward function, and $\gamma \in [0, 1)$ is a discount factor. Next, we will detail the main tuples in $\mathcal{M}$.

Augmented State. Instead of simply using the current states of the autonomous vehicle A and the target conventional vehicles $\mathbb{C}_{tar} = \{C_1, C_2, \ldots, C_6\}$ at time step t, we augment the future states of $\mathbb{C}_{tar}$ at time step $t+1$. Such augmentation enables the autonomous vehicle to receive possible changes of surrounding conventional vehicles to reduce the impact on them and avoid accidents. The current states refer to the latest historical states of the autonomous vehicle and the target conventional vehicles at time step t, and the future states refer to the predicted states of the target conventional vehicles at time step $t+1$, both derived from the enhanced perception module. Formally, the augmented state s_+^t at time step t is defined as $s_+^t = [h^t, \hat{f}^{t+1}]$. The current states h^t are formulated as follows:

$$h^t = [h_{A^t}, h_{C_1^t}, h_{C_2^t}, h_{C_3^t}, h_{C_4^t}, h_{C_5^t}, h_{C_6^t}] \tag{15}$$

where $h_{A^t} = [A^t.lat, A^t.lon, A^t.v, 0]$ and $h_{C_i^t} = [d_{lat}(C_i^t, A^t), d_{lon}(C_i^t, A^t), v(C_i^t, A^t), IF_{C_i}]$ based on Equation (7) and Equation (8). The future states $\hat{f}^{t+1}$ are formulated as follows:

$$\hat{f}^{t+1} = [\hat{f}'_{C_1^{t+1}}, \hat{f}'_{C_2^{t+1}}, \hat{f}'_{C_3^{t+1}}, \hat{f}'_{C_4^{t+1}}, \hat{f}'_{C_5^{t+1}}, \hat{f}'_{C_6^{t+1}}] \tag{16}$$

where $\hat{f}'_{C_i^{t+1}} = [\hat{d}_{lat}(C_i^{t+1}, A^t), \hat{d}_{lon}(C_i^{t+1}, A^t), \hat{v}(C_i^{t+1}, A^t), IF_{C_i}]$ based on Equation (13), and IF_{C_i} is a binary code used to indicate whether C_i is a phantom vehicle or not ($C_i \in \mathbb{C}_{tar}$).

Action. The maneuver of the autonomous vehicle follows the discrete-continuous hybrid action space, namely the parameterized action space [54], [55]. We define the action ac^t at time step t as follows:

$$ac^t = (A^t.b, A^t.a) \tag{17}$$

where $b \in \{ll, lr, lk\}$ is a discrete lateral lane change behavior and $a \in [-a', a']$ is a continuous longitudinal acceleration that ranges from $-a'$ to a'. The formulation indicates that each discrete $A^t.b$ has a corresponding continuous action-parameter $A^t.a$, which expresses our maneuver through the parameterized action space.

State Transition. After the autonomous vehicle plays an action ac^t at an augmented state s_+^t, we first update the current state $h_{A^{t+1}}$ of A at time step $t+1$ as follows:

$$h_{A^{t+1}} = [A^{t+1}.lat, A^{t+1}.lon, A^{t+1}.v, 0] = [A^t.lat + \overline{A^t.b}, A^t.lon + A^t.v\Delta t + 0.5A^t.a(\Delta t)^2, A^t.v + A^t.a\Delta t, 0] \tag{18}$$

where $\overline{A^t.b}$ equals to -1 ($A^t.b = ll$), $+1$ ($A^t.b = lr$) or 0 ($A^t.b = lk$), and Δt is the time interval between two consecutive time steps. Then the current state $h_{C_i^{t+1}}$ of each $C_i \in \mathbb{C}_{tar}$ at time step $t+1$ can be obtained via the sensor detection of the enhanced perception module. Accordingly, the future state $\hat{f}'_{C_i^{t+2}}$ of each $C_i \in \mathbb{C}_{tar}$ at time step $t+2$ can be predicted by executing LST-GAT model in the enhanced perception module. At last, s_+^t is updated to s_+^{t+1}. We note that once the autonomous vehicle reaches the destination of the road or causes a collision (e.g., a vehicle crash or hitting a road boundary), the augmented state will change to the terminal state s_T by default. In the scenario of autonomous driving, the state transition probability distribution $\mathcal{T}(s_+^{t+1}|s_+^t, ac^t)$ is non-deterministic due to the dynamic environment. Thus, we exploit a model-free reinforcement learning [56] algorithm.

Reward. Once the autonomous vehicle plays an action ac^t at an augmented state s_+^t, it will receive a reward as feedback, i.e., r^t. We construct a hybrid reward function to determine r^t, by considering four aspects: 1) safety, 2) efficiency, 3) comfort, and 4) impact, which are detailed in Section IV-C.

Based on the above PAMDP, we are required to find a policy π that tells the autonomous vehicle which is the best action to choose in each augmented state.

B. Deep Reinforcement Learning for Maneuver Decision

There are two main approaches to learning with parameterized actions: 1) alternate between optimizing the discrete actions and continuous action-parameters separately, e.g., P-QP [57]; or 2) collapse the parameterized action space into a continuous one, e.g., P-DDPG [58]. Both of them fail to fully exploit the structure present in parameterized action problems [55]. The former does not share information between the action and action-parameter policies, while the latter does not take into account which action-parameter is associated with which action. Recently, Parameterized Deep Q-Network (P-DQN) [54], [55] is proposed to directly learn the policy in the parameterized action space, which is a state-of-the-art method for solving PAMDP. However, this method shares the network structure for different inputs, which can lead to erroneous interactions between inputs with different scales, a.k.a. wrong weight sharing. Based on the optimization paradigm of P-DQN [54], we improve the structure and propose a novel network, called BP-DQN (Branched Parameterized Deep Q-Network), which handles different inputs separately. Next, we introduce the optimization paradigm based on P-DQN and detail the network structure of BP-DQN.

Optimization. The goal of our deep reinforcement learning is to learn a policy π with maximal expected γ-discounted cumulative reward, i.e., action-value function [59], as follows:

$$Q^*(s_+^t, ac^t) = \max_{\pi} \mathbb{E}\left[\sum_{t' \geq t} \gamma^{t'-t} r(s_+^{t'}, ac^{t'}) | \pi\right] \tag{19}$$

where $\gamma \in [0, 1)$ is a discount factor. The optimal policy is obtained by solving the optimal action-value function Q^*. We adopt P-DQN [54] as the reinforcement learning paradigm

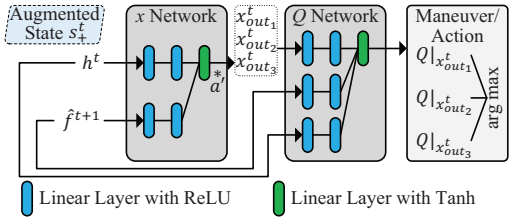

Fig. 6. Network Structure of BP-DQN

to solve our PAMDP, which defines the Bellman equation to estimate Q^* as follows:

$$\begin{aligned} Q(s_+^t, ac^t) &:= Q(s_+^t, A^t.b, A^t.a) = \mathbb{E}_{s_+^{t+1}}\Big[r(s_+^t, ac^t) + \\ &\gamma \max_{b \in \{ll, lr, lk\}} \sup_{a \in [-a', a']} Q(s_+^{t+1}, A^{t+1}.b, A^{t+1}.a) \Big| s_+^t, A^t.b, A^t.a\Big] \end{aligned} \tag{20}$$

To avoid the intractable calculation of sup, it states that when Q is fixed, we can view $\arg\sup_{a \in [-a', a']} Q(s_+^{t+1}, A^{t+1}.b, A^{t+1}.a)$ as a function $x(s_+^{t+1}, A^{t+1}.b)$. Then we can rewrite the above Bellman equation as follows:

$$\begin{aligned} Q(s_+^t, ac^t) &:= Q(s_+^t, A^t.b, A^t.a) = \mathbb{E}_{s_+^{t+1}}\Big[r(s_+^t, ac^t) + \\ &\gamma \max_{b \in \{ll, lr, lk\}} Q(s_+^{t+1}, A^{t+1}.b, x(s_+^{t+1}, A^{t+1}.b)) \Big| s_+^t, A^t.b, A^t.a\Big] \end{aligned} \tag{21}$$

Then we can use a deep neural network $Q(s_+^t, A^t.b, A^t.a; \theta_Q)$ to approximate $Q(s_+^t, A^t.b, A^t)$, where θ_Q is the learnable weights. Moreover, we approximate $x(s_+^t, A^t.b)$ using a deterministic policy network $x(s_+^t, A^t.b; \theta_x)$, where θ_x is the learnable weights [56]. Accordingly, it is easy to apply the standard DQN [59] approach of minimizing the mean-squared Bellman error to update Q network using mini-batches sampled from the replay buffer $\mathcal{B}$, as follows:

$$\begin{aligned} &\mathcal{L}_2 = \mathbb{E}_{(s_+^t, A^t.b, A^t, r(s_+^t, ac^t), s_+^{t+1}) \sim \mathcal{B}}\Big[\frac{1}{2}(y - Q(s_+^t, A^t.b, A^t.a; \theta_Q))^2\Big] \\ &y = r(s_+^t, ac^t) + \gamma \max_{b \in \{ll, lr, lk\}} Q(s_+^{t+1}, A^{t+1}.b, x(s_+^{t+1}, A^{t+1}.b; \theta_x); \theta_Q) \end{aligned} \tag{22}$$

The loss of x network is given by the negative sum [56] of Q values with fixed θ_Q, as follows:

$$\mathcal{L}_3 = \mathbb{E}_{s_+^t \sim \mathcal{B}}\Big[-\sum_{b \in \{ll, lr, lk\}} Q(s_+^t, A^t.b, x(s_+^t, A^t.b; \theta_x); \theta_Q)\Big] \tag{23}$$

Following $\mathcal{L}_2$, $\mathcal{L}_3$ and the standard training process used in P-DQN [54], we acquire the optimal action-value function Q^*. Then the optimal policy is able to select the best action with $\arg\max_{ac^t \in \mathcal{A}} Q^*(s_+^t, ac^t)$ at time step t.

Network Structure. The basic structure of P-DQN [54] consists of two networks, each of which is a single-branch computation for some inputs. In contrast to P-DQN, the structure of our BP-DQN consists of two networks (x and Q), each with multiple computational branches to compute different inputs separately, as shown in Figure 6. Using BP-DQN can effectively avoid erroneous weight sharing between inputs of different scales and improve the performance of P-DQN. Next, we introduce x network and Q network of BP-DQN.

The input of x network is the augmented state $s_+^t = [h^t, \hat{f}^{t+1}]$, and the output consists of three longitudinal accelerations corresponding to three lateral lane change behaviors,

i.e., $x^t_{out} = [A^t.a|_{A^t.b=ll}, A^t.a|_{A^t.b=lr}, A^t.a|_{A^t.b=lk}]$. We first process h^t and $\hat{f}^{t+1}$ respectively in two branches, then merge their intermediate results and calculate x^t_{out}. For $h^t \in \mathbb{R}^{4\times 7}$ and $\hat{f}^{t+1} \in \mathbb{R}^{4\times 6}$, we calculate their intermediate vectors, as follows:

$$\begin{aligned} h_{h^t} &= ReLU(\phi_6 ReLU(\phi_5 h^t + b_5) + b_6) \\ f_{\hat{f}^{t+1}} &= ReLU(\phi_8 ReLU(\phi_7 \hat{f}^{t+1} + b_7) + b_8) \end{aligned} \tag{24}$$

where $\phi_5 \in \mathbb{R}^{D_{\phi_5}\times 4}$, $\phi_6 \in \mathbb{R}^{D_{\phi_5}}$, $\phi_7 \in \mathbb{R}^{D_{\phi_7}\times 4}$, and $\phi_8 \in \mathbb{R}^{D_{\phi_7}}$ are the linear transformations, and b_5, b_6, b_7, b_8 are their biases. Then we calculate x^t_{out} as follows:

$$x^t_{out} = a' * Tanh(\phi_9[h_{h^t} \parallel f_{\hat{f}^{t+1}}] + b_9) \tag{25}$$

where a' is the is the boundary of acceleration, $\phi_9 \in \mathbb{R}^{3\times 13}$ and b_9 is the bias. Thus, three longitudinal accelerations inside x^t_{out} will be limited from $-a'$ to a', following the action range of PAMDP.

The inputs of Q network consists of the augmented state $s^t_+ = [h^t, \hat{f}^{t+1}]$ and the output x^t_{out} from x network. The outputs of Q network are three Q values corresponding to three longitudinal accelerations inside x^t_{out}, i.e., $Q^t_{out} = [Q|_{x^t_{out_1}}, Q|_{x^t_{out_2}}, Q|_{x^t_{out_3}}]$. Similar to x network, We first process h^t, $\hat{f}^{t+1}$, and x^t_{out} respectively in three branches, then merge their intermediate results and calculate Q^t_{out}. For $h^t \in \mathbb{R}^{4\times 7}$, $\hat{f}^{t+1} \in \mathbb{R}^{4\times 6}$, and $x^t_{out} \in \mathbb{R}^3$, we calculate their intermediate vectors as follows:

$$\begin{aligned} h'_{h^t} &= ReLU(\phi_{11} ReLU(\phi_{10} h^t + b_{10}) + b_{11}) \\ f'_{\hat{f}^{t+1}} &= ReLU(\phi_{13} ReLU(\phi_{12} \hat{f}^{t+1} + b_{12}) + b_{13}) \\ x'_{x^t_{out}} &= ReLU(\phi_{15} ReLU(\phi_{14} x^t_{out} + b_{14}) + b_{15}) \end{aligned} \tag{26}$$

where $\phi_{10} \in \mathbb{R}^{D_{\phi_{10}}\times 4}$, $\phi_{11} \in \mathbb{R}^{D_{\phi_{10}}}$, $\phi_{12} \in \mathbb{R}^{D_{\phi_{12}}\times 4}$, $\phi_{13} \in \mathbb{R}^{D_{\phi_{12}}}$, $\phi_{14} \in \mathbb{R}^{D_{\phi_{14}}\times 3}$, and $\phi_{15} \in \mathbb{R}^{3\times D_{\phi_{14}}}$ are the linear transformations, and b_{10}, b_{11}, b_{12}, b_{13}, b_{14}, and b_{15} are their biases. Then we calculate Q^t_{out} as follows:

$$Q^t_{out} = \phi_{16}[h'_{h^t} \parallel f'_{\hat{f}^{t+1}} \parallel x'_{x^t_{out}}] + b_{16} \tag{27}$$

where $\phi_{16} \in \mathbb{R}^{3\times 16}$ and b_{16} is the bias. Q^t_{out} indicates three expected γ-discounted cumulative rewards w.r.t. three longitudinal accelerations inside x^t_{out}. Thus, the policy selects the best action with $\arg\max_{x^t_{out_i} \in \{x^t_{out_1}, x^t_{out_2}, x^t_{out_3}\}} Q^t_{out}$, where $x^t_{out_1} = A^t.a|_{A^t.b=ll}$, $x^t_{out_2} = A^t.a|_{A^t.b=lr}$, and $x^t_{out_3} = A^t.a|_{A^t.b=lk}$.

C. Hybrid Reward Function

The hybrid reward function serves as an exploration signal to teach the autonomous vehicle to learn the optimal policy. In this work, a good action of the autonomous vehicle should be safe, efficient, comfortable, and with minimal negative impacts on its surrounding conventional vehicles. Therefore, we construct a hybrid reward function considering four aspects: 1) safety, 2) efficiency, 3) comfort, and 4) impact, as follows:

$$r^t = w_1 r^t_1 + w_2 r^t_2 + w_3 r^t_3 + w_4 r^t_4 \tag{28}$$

where w_1, w_2, w_3, and w_4 are four tunable coefficients to adjust the importance of safety, efficiency, comfort, and impact, respectively. Next, we define the reward values for safety (r^t_1), efficiency (r^t_1), comfort (r^t_3), and impact (r^t_4).

Safety. Time to collision (TTC), as a widely used safety indicator, represents the time span left before a collision if two vehicles maintain their current velocities [60]. Following the standard definition [11], we mainly consider TTC of the autonomous vehicle with its front conventional vehicle, after the autonomous vehicle plays an action, i.e., $TTC^{t+1} = \frac{d_{lat}(C_2^{t+1}, A^{t+1})}{-v(C_2^{t+1}, A^{t+1})}$, where $v(C_2^{t+1}, A^{t+1}) < 0$ (otherwise TTC^{t+1} is invalid). Further, if the autonomous vehicle causes a collision, we will use a low reward to reflect its lack of safety. Thus, the safety reward value $r^t_1 \in [-3, 0]$ is defined as follows:

$$r^t_1 = \begin{cases} -3 & collision \\ \max\Big(-3, \log\big(\frac{TTC^{t+1}}{\mathcal{G}}\big)\Big) & 0 \le TTC^{t+1} < \mathcal{G} \\ 0 & otherwise \end{cases} \tag{29}$$

where $collision$ refers to a vehicle crash or hitting a road boundary, $\mathcal{G}$ is a scaling threshold, and $\max$ calculation clips $\log(\frac{TTC^{t+1}}{\mathcal{G}})$ to $[-3, 0)$. The autonomous vehicle usually causes a collision when $\log(\frac{TTC^{t+1}}{\mathcal{G}}) \le -3$. If C_2 is a constructed phantom vehicle, we mask its TTC and only consider whether the autonomous vehicle causes a collision.

Efficiency. The longitudinal velocity of the autonomous vehicle directly reflects its driving efficiency [14]. Thus, the efficiency reward value $r^t_2 \in [0, 1]$ is defined as $r^t_2 = (A^{t+1}.v - v_{min})/(v_{max} - v_{min})$.

Comfort. Jerk, 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 [61]. The larger Jerk is, the more uncomfortable passengers will feel [11]. Accordingly, the comfort reward value $r^t_3 \in [-1, 0]$ is defined as $r^t_3 = -|A^t.a - A^{t-1}.a|/2a'$.

Impact. The deceleration or lane change behavior of the autonomous vehicle could affect its surrounding conventional vehicles, reducing their driving efficiency and comfort, especially for some inappropriate behaviors, e.g., hard braking and forced lane change [14]. In order to reduce the impact of the autonomous vehicle, we design an impact reward value to measure the degree that the autonomous vehicle forces the surrounding conventional vehicles to decelerate. According to the analysis of existing studies [14], [15], whether a vehicle decelerates or changes lanes, it only affects the vehicle behind it after performing the action. Therefore, after the autonomous vehicle plays an action, we record the deceleration of its rear conventional vehicle to measure the impact of the autonomous vehicle. Specifically, the impact reward value $r^t_4 \in [-1, 0]$ is defined as follows:

$$r^t_4 = \begin{cases} \frac{C_5^{t+1}.v - C_5^t.v}{2a' * \Delta t} & C_5^t.v - C_5^{t+1}.v > v_{thr} \\ 0 & otherwise \end{cases} \tag{30}$$

where v_{thr} is a threshold of velocity used to measure whether the autonomous vehicle affects the rear conventional vehicle C_5, and $2a' * \Delta t$ refers to the largest velocity change between two consecutive time steps. The impact reward value only works when C_5 decelerates greater than v_{thr} at time step $t+1$. If C_5 is a constructed phantom vehicle, we will set r^t_4 as 0 to mask its impact.

V. Experiments

A. Experimental Settings

Datasets. As it requires interaction between the autonomous and conventional vehicles, most of the experiments are conducted in a simulated environment, called SUMO (Simulation of Urban MObility)[2]. SUMO is an open-source, highly portable, and microscopic simulator that operates at each vehicle level and allows users to import road networks and define corresponding requests [62]. The interaction between SUMO and our framework *HEAD* is established by using TraCI (Traffic Control Interface)[3], which allows to retrieve values of simulated objects and to manipulate their behaviors online. In SUMO, we simulate a straight six-lane road of length $3km$, where the width of each lane wid_l is $3.2m$. There are many SUMO-controlled conventional vehicles and one *HEAD*-controlled autonomous vehicle traveling on the road. The density of the road is set as 180 vehicles per kilometer, which is appropriate for autonomous driving studies. The autonomous vehicle is initialized at the road origin on a random lane. We set the traffic restrictions as $v_{min} = 5km/h \approx 1.39m/s$, $v_{max} = 90km/h = 25m/s$, and $a' = 3m/s^2$, following the settings in the previous works [11], [14], [63]. We denote an episode as the autonomous vehicle driving from the origin to the destination of the road, or driving from the origin to the location of a collision. We prepare $4,000$ episodes for training, and 500 episodes for testing. Each episode is initialized randomly to guarantee the variance.

Furthermore, we evaluate our LST-GAT model on a dataset constructed by merging two real-world datasets: NGSIM US-101 [64] and I-80 [65]. The merged dataset, called REAL, consists of trajectories of conventional vehicles traveling on a $1.14km$-length highway segment with six straight lanes, which is produced by the standard preprocessing used in the previous work [14]. We further split REAL into training and test sets with a splitting ratio of $4:1$. Since the distributions of SUMO and REAL are similar, we can train our LST-GAT model on REAL and use it to fulfill the state prediction in SUMO.

Implementation Details. We implement the enhanced perception and maneuver decision modules in *HEAD* as follows:
(**1**) Enhanced perception module. For the phantom vehicle construction, the sensor detection range (or radius) $\mathcal{R}$ of the autonomous vehicle is set to $100m$, and the number of historical time steps z is set to 5, following the settings used in the previous works [14], [27]. Although SUMO provides global states for all conventional vehicles, we use the geometry [66] to simulate the sensor limitations in the real scenario of autonomous driving. For the network structure of LST-GAT model, we set the corresponding dimensions as: $D_{\phi_1} = 64$, $D_{\phi_3} = 64$, and $D_l = 64$. In addition, we train LST-GAT model by using Adam optimizer [67] for 15 epochs with a learning rate of 0.001 and a batch size of 64 by default.
(**2**) Maneuver decision module. For the network structure of BP-DQN, we set the corresponding dimensions as: $D_{\phi_5} = 64$, $D_{\phi_7} = 64$, $D_{\phi_{10}} = 64$, $D_{\phi_{12}} = 64$, $D_{\phi_{14}} = 64$. For the hybrid reward function, the scaling threshold $\mathcal{G}$ is set as 4, and the threshold of velocity v_{thr} is set to $0.5m/s$, following the settings used in the previous works [11], [68]. Further, we set the tunable coefficients used in the hybrid reward function as: $w_1 = 0.9$, $w_2 = 0.8$, $w_3 = 0.6$, $w_4 = 0.2$, which are evaluated in Section V-E. For the optimization of BP-DQN, the discount factor γ is set to 0.9, the size of the replay buffer is set to $20,000$ by default. We train our BP-DQN using Adam optimizer [67] for $4,000$ episodes with a scheduled learning rate of 0.001 and a batch size of 64, following the settings used in the previous works [15], [55]. Similar to DDPG [56], we add two target networks for Q and x networks and use the soft updates strategy with the ratio of 0.01 for training stability. Our experimental results are reported based on the above settings, unless expressly specified.

Baselines. We compare our *HEAD* with several representative baselines for autonomous driving, including two traditional methods, a deep reinforcement learning-based method, and a prediction-and-search method, as follows:
(**1**) IDM-LC [8], [69]. Traditional intelligent driver model with lane-changing model for decision making.
(**2**) ACC-LC [6]–[8]. Traditional adaptive cruise control model with lane-changing model for decision making.
(**3**) DRL-SC [10]. Deep reinforcement learning model with safety check for decision making.
(**4**) TP-BTS [14]. Trajectory prediction model with behavior tree search algorithm for decision making.

Variants. To evaluate each component of our framework, we perform ablation studies with the following variants of *HEAD*:
(**1**) *HEAD*-w/o-PVC. We remove the strategy of phantom vehicle construction in the enhanced perception module, and the historical states of unobservable conventional vehicles are filled with zero states.
(**2**) *HEAD*-w/o-LST-GAT. We remove our LST-GAT model in the enhanced perception module, and only use the current observable states for maneuver decisions.
(**3**) *HEAD*-w/o-BP-DQN. We replace our BP-DQN with the vanilla P-DQN [54] in the maneuver decision module.
(**4**) *HEAD*-w/o-IMP. We remove the impact reward value in the maneuver decision module, and only consider the safety, efficiency, and comfort reward values.

Other Compared Methods. For LST-GAT model in the enhanced perception module, we compare it against several state-of-the-art trajectory prediction methods modified for our state prediction task, as follows:
(**1**) LSTM-MLP [26]. Vanilla LSTM with multilayer perceptron network for state prediction.
(**2**) ED-LSTM [37]. Encoder-decoder LSTM network for state prediction.
(**3**) GAS-LED [14]. Global attention and state sharing LSTM encoder-decoder network for state prediction.

Furthermore, for BP-DQN in the maneuver decision module, we compare it against three state-of-the-art reinforcement learning-based methods for solving our PAMDP, as follows:

[2]https://www.eclipse.org/sumo/
[3]https://sumo.dlr.de/docs/TraCI.html

TABLE I
END-TO-END PERFORMANCE OF BASELINES AND *HEAD* IN SUMO

Method	Macroscopic			Microscopic			
	Avg DT-A (s)	Avg DT-C (s)	Avg #-CA	Min TTC-A (s)	Avg V-A (m/s)	Avg J-A (m/s^2)	Avg D-CA (m/s)
IDM-LC	143.9	167.5	23.9	3.42	20.84	0.47	0.22
ACC-LC	142.4	167.3	25.4	3.48	21.06	0.48	0.24
DRL-SC	139.6	165.4	24.3	3.67	21.48	0.45	0.23
TP-BTS	141.2	164.2	21.5	3.31	21.24	0.44	0.21
HEAD	**134.5**	**161.3**	**16.7**	**3.85**	**22.30**	**0.37**	**0.18**

(**1**) P-QP [57]. Parameterized Q-learning with parameter update algorithm for PAMDP.
(**2**) P-DDPG [58]. Parameterized deep deterministic policy gradients algorithm for PAMDP.
(**3**) P-DQN [54]. Parameterized deep Q-network for PAMDP.
Environments. All of the experiments are run on an an Ubuntu Server with an Intel(R) Xeon(R) Silver 4214 CPU @ 2.20GHz, and NVIDIA GeForce RTX 3080 GPU.

B. End-to-End Evaluation

We study the end-to-end performance of *HEAD* by comparing it against several baselines (IDM-LC, ACC-LC, DRL-SC, and TP-BTS). We conduct 500 test episodes in SUMO[4], and measure the effectiveness from both macroscopic and microscopic aspects.
Macroscopic Effectiveness. We design the macroscopic metrics as follows:
(**1**) Average driving time of the autonomous vehicle (AvgDT-A). We record the end-to-end driving time of the autonomous vehicle traveling through the road ($3km$). The smaller value of AvgDT-A indicates that the autonomous vehicle has higher driving efficiency.
(**2**) Average driving time of conventional vehicles (AvgDT-C). We record the end-to-end driving time of the conventional vehicles within 100 meters behind the autonomous vehicle traveling through the road ($3km$). The smaller value of AvgDT-C indicates higher traffic flow efficiency.
(**3**) Average number of times that the autonomous vehicle affects its rear conventional vehicle (Avg#-CA). We record the velocity change of the conventional vehicle behind the autonomous vehicle, if it decelerates greater than $0.5m/s$ from t to $t+1$, our counter will record it. The smaller value of Avg#-CA indicates that the autonomous vehicle has less impact on traffic flow.

We report AvgDT-A, AvgDT-C, and Avg#-CA in Table I. As shown, *HEAD* achieves the shortest AvgDT-A and AvgDT-C, and has the least Avg#-CA. These results demonstrate that *HEAD* not only enables the autonomous vehicle to have high driving efficiency, but also improves traffic flow efficiency.
Microscopic Effectiveness. We design the microscopic metrics as follows:
(**1**) Minimum time to collision of the autonomous vehicle (MinTTC-A). We record TTC of the autonomous vehicle (in Section IV-C). The larger value of MinTTC-A indicates that the autonomous vehicle is safer.

[4]All methods do not cause any collisions (or accidents) in the test episodes.

TABLE II
ABLATION STUDY OF *HEAD*-VARIANTS AND *HEAD* IN SUMO

Method	Macroscopic			Microscopic			
	Avg DT-A (s)	Avg DT-C (s)	Avg #-CA	Min TTC-A (s)	Avg V-A (m/s)	Avg J-A (m/s^2)	Avg D-CA (m/s)
HEAD-w/o-PVC	139.2	163.4	18.5	3.76	21.54	0.42	0.20
HEAD-w/o-LST-GAT	140.7	164.6	19.7	3.69	21.32	0.39	0.21
HEAD-w/o-BP-DQN	137.1	162.2	17.3	3.82	21.87	0.40	0.19
HEAD-w/o-IMP	135.5	163.8	22.6	3.73	22.13	0.41	0.22

(**2**) Average velocity of the autonomous vehicle (AvgV-A). We record the velocity of the autonomous vehicle in all test episodes. The larger value of AvgV-A indicates that the autonomous vehicle is faster.
(**3**) Average Jerk of the autonomous vehicle (AvgJ-A). We record Jerk of the autonomous vehicle (in Section IV-C). The smaller value of AvgJ-A indicates that the autonomous vehicle is more comfortable.
(**4**) Average deceleration of the conventional vehicle behind the autonomous vehicle (AvgD-CA). We record the deceleration of the conventional vehicle behind the autonomous vehicle car in all test episodes. The smaller value of AvgD-CA indicates that the autonomous vehicle has less impact on its rear conventional vehicles.

We report MinTTC-A, AvgV-A, AvgJ-A, and AvgD-CA in Table I. We can see that *HEAD* has the longest MinTTC-A, the highest AvgV-A, and the lowest AvgJ-A and AvgD-CA, demonstrating that *HEAD* enables the autonomous vehicle to perform safe and comfortable maneuvers with high velocity and minimal impact on surrounding conventional vehicles.

C. Evaluation of Enhanced Perception Module

We evaluate the performance of the enhanced perception module in *HEAD* by conducting an ablation study and a break-down evaluation. The ablation study is conducted by comparing *HEAD* against several *HEAD*-variants (*HEAD*-w/o-PVC and *HEAD*-w/o-LST-GAT). The break-down evaluation is conducted by comparing LST-GAT model with three state prediction methods (LSTM-MLP, ED-LSTM, and GAS-LED).
Ablation Study. For *HEAD*-w/o-PVC and *HEAD*-w/o-LST-GAT, we measure their effectiveness as in Section V-B. We report their AvgDT-A, AvgDT-C, Avg#-CA, MinTTC-A, AvgV-A, AvgJ-A, and AvgD-CA in Table II. From the macroscopic aspects, *HEAD* achieves the shortest AvgDT-A and AvgDT-C, and has the least Avg#-CA, which proves that LST-GAT model with the strategy of phantom vehicle construction is benefit for both perception and decision of autonomous driving. From the microscopic aspects, we can see that *HEAD* achieves the longest MinTTC-A, the highest AvgV-A, and the lowest AvgJ-A and AvgD-CA, demonstrating that LST-GAT model with the strategy of phantom vehicle construction is useful for making safe, efficient, and comfortable decisions with minimal impact.
Break-down Evaluation. We take a break-down evaluation of our LST-GAT model by comparing it with LSTM-MLP,

TABLE III
ACCURACY OF COMPARED METHODS AND LST-GAT ON REAL

Metric	LSTM-MLP	ED-LSTM	GAS-LED	LST-GAT
MAE	0.62	0.47	0.35	**0.27**
MSE	0.13	0.08	0.05	**0.03**
RMSE	0.35	0.28	0.22	**0.17**

TABLE IV
EFFICIENCY OF COMPARED METHODS AND LST-GAT ON REAL

Metric	LSTM-MLP	ED-LSTM	GAS-LED	LST-GAT
TCT (h)	0.87	0.94	1.06	**0.83**
AvgIT (ms)	8.58	9.43	10.51	**3.76**

ED-LSTM, and GAS-LED from both accuracy and efficiency aspects on REAL. For the accuracy, we report their Mean absolute error (MAE), Mean squared error (MSE), and Root mean squared error (RMSE) for our one-step state prediction in Table III. For the efficiency, we report their Training convergence time (TCT) and Average inference time (AvgIT) in Table IV. As depicted, our proposed LST-GAT model outperforms all the other models, since it has lower MAE, MSE, and RMSE, and has shorter TCT and AvgIT.

D. Evaluation of Maneuver Decision Module

We evaluate the performance of the maneuver decision module in *HEAD* by conducting an ablation study and a break-down evaluation. The ablation study is conducted by comparing *HEAD* against several *HEAD*-variants (*HEAD*-w/o-BP-DQN and *HEAD*-w/o-IMP). The break-down evaluation is conducted by comparing our BP-DQN with three reinforcement learning-based methods (P-QP, P-DDPG, and P-DQN).

Ablation Study. For *HEAD*-w/o-BP-DQN and *HEAD*-w/o-IMP, we measure their effectiveness as in Section V-B. We report their AvgDT-A, AvgDT-C, Avg#-CA, MinTTC-A, AvgV-A, AvgJ-A, and AvgD-CA in Table II. From the macroscopic aspects, *HEAD* achieves the shortest AvgDT-A and AvgDT-C, and has the least Avg#-CA, which clearly proves that BP-DQN and the hybrid reward function is benefit for making appropriate decisions without affecting traffic flow efficiency. From the microscopic aspects, we can see that *HEAD* has the longest MinTTC-A, the highest AvgV-A, and the lowest AvgJ-A and AvgD-CA, proving that BP-DQN and the impact reward value not only improve the safety, efficiency, and comfort of the autonomous vehicle but also reduce the impact.

Break-down Evaluation. We take a break-down evaluation of our BP-DQN by comparing it with P-QP, P-DDPG, and P-DQN from both effectiveness and efficiency aspects in SUMO. For the effectiveness, we report their Minimum reward (MinR), Maximum reward (MaxR), and Average reward (AvgR) in Table V. For the efficiency, we report their Training convergence time (TCT) and Average inference time (AvgIT) in Table VI. As shown, our BP-DQN outperforms all the other reinforcement learning-based methods, since it has higher MinR, MaxR, and AvgR, and has shorter TCT and AvgIT.

E. Reward Shaping

For the hybrid reward function in the maneuver decision module, w_1, w_2, w_3, and w_4 correspond to the coefficients of r_1^t (safety), r_2^t (efficiency), r_3^t (comfort), and r_4^t (impact), respectively. To achieve better performance, we adopt the grid search [70] to determine them, which is shown in Table VII.

TABLE V
EFFECTIVENESS OF COMPARED METHODS AND BP-DQN IN SUMO

Metric	P-QP	P-DDPG	P-DQN	BP-DQN
MinR	-2.06	-1.73	-1.26	**-1.14**
MaxR	0.35	0.47	0.56	**0.59**
AvgR	0.17	0.29	0.36	**0.43**

TABLE VI
EFFICIENCY OF COMPARED METHODS AND BP-DQN IN SUMO

Metric	P-QP	P-DDPG	P-DQN	BP-DQN
TCT (h)	1.83	1.85	1.77	**1.74**
AvgIT (ms)	3.59	3.64	3.57	**3.48**

TABLE VII
EFFECT OF COEFFICIENTS IN HYBRID REWARD FUNCTION

Coefficient	Min	Max	Step	Best
w_1	0.5	1	0.1	**0.9**
w_2	0	1	0.2	**0.8**
w_3	0	1	0.2	**0.6**
w_4	0	0.5	0.1	**0.2**

VI. RELATED WORK

Decision-making for autonomous vehicles focuses on how to use the data provided by their sensors to make maneuver decisions [5], [9]. Traditional decision-making methods, e.g., Krauss [71], IDM [69], ACC [6], [7], and LC [8], design a set of rule-matching algorithms to perform velocity change and lane change behaviors, which require expert experience and manual tuning, leading to poor generalizability. Afterward, the autonomous driving community attempts to utilize reinforcement learning-based methods to make maneuver decisions [9], [72]. Existing reinforcement learning-based models, e.g., DRL-SC [10], AD-DDPG [11], MCTS-DRL [12], and EA-DQN [13], mainly optimize the driving safety, efficiency, and comfort of autonomous vehicles, leaving the impact on other surrounding vehicles. Recently, several prediction-and-decision frameworks [14], [73], [74] are proposed to first proactively perceive possible changes of surrounding vehicles' states and then make maneuver decisions more wisely. However, they cannot quantify the impact of the autonomous vehicle on surrounding vehicles, and ignore the sensor limitations in their prediction phases, leading to poor applicability.

VII. CONCLUSION

In this work, we propose a perception-and-decision framework, called *HEAD*, to enable the autonomous vehicle to perform safe, efficient, and comfortable maneuvers with minimal impact on surrounding vehicles. Experiments confirm the superiority of *HEAD* over state-of-the-art approaches.

ACKNOWLEDGMENT

This work is partially supported by NSFC (No. 61972069, 61836007, 61832017, 62272086), Shenzhen Municipal Science and Technology R&D Funding Basic Research Program (JCYJ20210324133607021), and Municipal Government of Quzhou under Grant (No. 2021D022, 2022D037).

REFERENCES

[1] D. Schrank, L. Albert, B. Eisele, and T. Lomax, "2021 urban mobility report," Texas A&M Transportation Institute, Tech. Rep., 2021.

[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, 2016.

[3] Z. Wang, M. Lu, X. Yuan, J. Zhang, and H. Van De Wetering, "Visual traffic jam analysis based on trajectory data," *IEEE transactions on visualization and computer graphics*, vol. 19, no. 12, pp. 2159–2168, 2013.

[4] Y. Sugiyama, M. Fukui, M. Kikuchi, K. Hasebe, A. Nakayama, K. Nishinari, S.-i. Tadaki, and S. Yukawa, "Traffic jams without bottlenecks—experimental evidence for the physical mechanism of the formation of a jam," *New journal of physics*, vol. 10, no. 3, p. 033001, 2008.

[5] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE access*, vol. 8, pp. 58 443–58 469, 2020.

[6] V. Milanés and S. E. Shladover, "Modeling cooperative and autonomous adaptive cruise control dynamic responses using experimental data," *Transportation Research Part C: Emerging Technologies*, vol. 48, pp. 285–300, 2014.

[7] L. Xiao, M. Wang, and B. Van Arem, "Realistic car-following models for microscopic simulation of adaptive and cooperative adaptive cruise control vehicles," *Transportation Research Record*, vol. 2623, no. 1, pp. 1–9, 2017.

[8] J. Erdmann, "Sumo's lane-changing model," in *Modeling Mobility with Open Data*. Springer, 2015, pp. 105–123.

[9] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[10] S. Nageshrao, H. E. Tseng, and D. Filev, "Autonomous highway driving using deep reinforcement learning," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. IEEE, 2019, pp. 2326–2331.

[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] C.-J. Hoel, K. Driggs-Campbell, K. Wolff, L. Laine, and M. J. Kochenderfer, "Combining planning and deep reinforcement learning in tactical decision making for autonomous driving," *IEEE transactions on intelligent vehicles*, vol. 5, no. 2, pp. 294–305, 2019.

[13] E. Leurent and J. Mercat, "Social attention for autonomous decision-making in dense traffic," *arXiv preprint arXiv:1911.12250*, 2019.

[14] S. Liu, H. Su, Y. Zhao, K. Zeng, and K. Zheng, "Lane change scheduling for autonomous vehicle: A prediction-and-search framework," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 3343–3353.

[15] Z. Xu, S. Liu, Z. Wu, X. Chen, K. Zeng, K. Zheng, and H. Su, "Patrol: A velocity control framework for autonomous vehicle via spatial-temporal reinforcement learning," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 2271–2280.

[16] S. Mozaffari, O. Y. Al-Jarrah, M. Dianati, P. Jennings, and A. Mouzakitis, "Deep learning-based vehicle behavior prediction for autonomous driving applications: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 1, pp. 33–47, 2020.

[17] N. Kehtarnavaz, N. Groswold, K. Miller, and P. Lascoe, "A transportable neural-network approach to autonomous vehicle following," *IEEE Transactions on Vehicular Technology*, vol. 47, no. 2, pp. 694–702, 1998.

[18] J. Jiang and A. Astolfi, "Lateral control of an autonomous vehicle," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 228–237, 2018.

[19] J. Li, B. Dai, X. Li, X. Xu, and D. Liu, "A dynamic bayesian network for vehicle maneuver prediction in highway driving scenarios: Framework and verification," *Electronics*, vol. 8, no. 1, p. 40, 2019.

[20] W. Shi, M. B. Alawieh, X. Li, and H. Yu, "Algorithm and hardware implementation for visual perception system in autonomous vehicle: A survey," *Integration*, vol. 59, pp. 148–156, 2017.

[21] J. Yoo and R. Langari, "A predictive perception model and control strategy for collision-free autonomous driving," *IEEE transactions on intelligent transportation systems*, vol. 20, no. 11, pp. 4078–4091, 2018.

[22] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla, "A systematic review of perception system and simulators for autonomous vehicles research," *Sensors*, vol. 19, no. 3, p. 648, 2019.

[23] W. Xu, J. Wei, J. M. Dolan, H. Zhao, and H. Zha, "A real-time motion planner with trajectory optimization for autonomous vehicles," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 2061–2067.

[24] L. R. Medsker and L. Jain, "Recurrent neural networks," *Design and Applications*, vol. 5, pp. 64–67, 2001.

[25] 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.

[26] F. Altché and A. de La Fortelle, "An lstm network for highway trajectory prediction," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 353–359.

[27] 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.

[28] K. Messaoud, I. Yahiaoui, A. Verroust-Blondet, and F. Nashashibi, "Non-local social pooling for vehicle trajectory prediction," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 975–980.

[29] R. Roriz, J. Cabral, and T. Gomes, "Automotive lidar technology: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6282–6297, 2022.

[30] S. Royo and M. Ballesta-Garcia, "An overview of lidar imaging systems for autonomous vehicles," *Applied sciences*, vol. 9, no. 19, p. 4093, 2019.

[31] B. Fu, Y. Wang, X. Ding, Y. Jiao, L. Tang, and R. Xiong, "Lidar-camera calibration under arbitrary configurations: Observability and methods," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 6, pp. 3089–3102, 2019.

[32] Y. Li and J. Ibanez-Guzman, "Lidar for autonomous driving: The principles, challenges, and trends for automotive lidar and perception systems," *IEEE Signal Processing Magazine*, vol. 37, no. 4, pp. 50–61, 2020.

[33] F. Baselice, G. Ferraioli, G. Matuozzo, V. Pascazio, and G. Schirinzi, "3d automotive imaging radar for transportation systems monitoring," in *2014 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems Proceedings*. IEEE, 2014, pp. 1–5.

[34] Y. Shen and W. Q. Yan, "Blind spot monitoring using deep learning," in *2018 International Conference on Image and Vision Computing New Zealand (IVCNZ)*. IEEE, 2018, pp. 1–5.

[35] J. Liu, X. Mao, Y. Fang, D. Zhu, and M. Q.-H. Meng, "A survey on deep-learning approaches for vehicle trajectory prediction in autonomous driving," in *2021 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2021, pp. 978–985.

[36] K. Messaoud, I. Yahiaoui, A. Verroust-Blondet, and F. Nashashibi, "Attention based vehicle trajectory prediction," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 1, pp. 175–185, 2020.

[37] 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.

[38] K. Messaoud, N. Deo, M. M. Trivedi, and F. Nashashibi, "Trajectory prediction for autonomous driving based on multi-head attention with joint agent-map representation," in *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021, pp. 165–170.

[39] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 01, 2020, pp. 1234–1241.

[40] M. Qiu, P. Zhao, K. Zhang, J. Huang, X. Shi, X. Wang, and W. Chu, "A short-term rainfall prediction model using multi-task convolutional neural networks," in *2017 IEEE international conference on data mining (ICDM)*. IEEE, 2017, pp. 395–404.

[41] S. Liu, Z. Xu, H. Ren, T. He, B. Han, J. Bao, K. Zheng, and Y. Zheng, "Detecting loaded trajectories for hazardous chemicals transportation," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2022, pp. 3294–3306.

[42] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 922–929.

[43] C. Zhang, J. James, and Y. Liu, "Spatial-temporal graph attention networks: A deep learning approach for traffic forecasting," *IEEE Access*, vol. 7, pp. 166 246–166 256, 2019.

[44] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[45] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[46] M. Fu, T. Zhang, W. Song, Y. Yang, and M. Wang, "Trajectory prediction-based local spatio-temporal navigation map for autonomous driving in dynamic highway environments," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[47] C. Wissing, T. Nattermann, K.-H. Glander, and T. Bertram, "Trajectory prediction for safety critical maneuvers in automated highway driving," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 131–136.

[48] S. Dai, L. Li, and Z. Li, "Modeling vehicle interactions via modified lstm models for trajectory prediction," *IEEE Access*, vol. 7, pp. 38 287–38 296, 2019.

[49] H.-T. Cheng, C.-H. Chao, J.-D. Dong, H.-K. Wen, T.-L. Liu, and M. Sun, "Cube padding for weakly-supervised saliency prediction in 360 videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1420–1429.

[50] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *The world wide web conference*, 2019, pp. 2022–2032.

[51] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "Kgat: Knowledge graph attention network for recommendation," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 950–958.

[52] V. Kosaraju, A. Sadeghian, R. Martín-Martín, I. Reid, H. Rezatofighi, and S. Savarese, "Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[53] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[54] J. Xiong, Q. Wang, Z. Yang, P. Sun, L. Han, Y. Zheng, H. Fu, T. Zhang, J. Liu, and H. Liu, "Parametrized deep q-networks learning: Reinforcement learning with discrete-continuous hybrid action space," *arXiv preprint arXiv:1810.06394*, 2018.

[55] C. J. Bester, S. D. James, and G. D. Konidaris, "Multi-pass q-networks for deep reinforcement learning with parameterised action spaces," *arXiv preprint arXiv:1905.04388*, 2019.

[56] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[57] W. Masson, P. Ranchod, and G. Konidaris, "Reinforcement learning with parameterized actions," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[58] M. Hausknecht and P. Stone, "Deep reinforcement learning in parameterized action space," *arXiv preprint arXiv:1511.04143*, 2015.

[59] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[60] L. Evans, *Traffic safety and the driver*. Science Serving Society, 1991.

[61] 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.

[62] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *2018 21st international conference on intelligent transportation systems (ITSC)*. IEEE, 2018, pp. 2575–2582.

[63] 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.

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

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

[66] J. Yan, H. Jiao, W. Pu, C. Shi, J. Dai, and H. Liu, "Radar sensor network resource allocation for fused target tracking: a brief review," *Information Fusion*, 2022.

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

[68] X. Qu, Y. Yu, M. Zhou, C.-T. Lin, and X. Wang, "Jointly dampening traffic oscillations and improving energy consumption with electric, connected and automated vehicles: a reinforcement learning based approach," *Applied Energy*, vol. 257, p. 114030, 2020.

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

[70] I. Syarif, A. Prugel-Bennett, and G. Wills, "Svm parameter optimization using grid search and genetic algorithm to improve classification performance," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 14, no. 4, pp. 1502–1509, 2016.

[71] S. Krauß, P. Wagner, and C. Gawron, "Metastable states in a microscopic model of traffic flow," *Physical Review E*, vol. 55, no. 5, p. 5597, 1997.

[72] S. Aradi, "Survey of deep reinforcement learning for motion planning of autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[73] C. Hubmann, M. Becker, D. Althoff, D. Lenz, and C. Stiller, "Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 1671–1678.

[74] S. Noh, "Decision-making framework for autonomous driving at road intersections: Safeguarding against collision, overly conservative behavior, and violation vehicles," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 4, pp. 3275–3286, 2018.