Target-Oriented Maneuver Decision for Autonomous Vehicle: A Rule-Aided Reinforcement Learning Framework

Ximu Zeng
University of Electronic Science and Technology of China
Chengdu, China
ximuzeng@std.uestc.edu.cn

Quanlin Yu
University of Electronic Science and Technology of China
Chengdu, China
quanlin.yu@std.uestc.edu.cn

Shuncheng Liu
University of Electronic Science and Technology of China
Chengdu, China
liushuncheng@std.uestc.edu.cn

Yuyang Xia
University of Electronic Science and Technology of China
Chengdu, China
xiayuyang@std.uestc.edu.cn

Han Su*
Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China
Quzhou, China
hansu@uestc.edu.cn

Kai Zheng*†
University of Electronic Science and Technology of China
Chengdu, China
zhengkai@uestc.edu.cn

ABSTRACT

Autonomous driving systems (ADSs) have the potential to revolutionize transportation by improving traffic safety and efficiency. As the core component of ADSs, maneuver decision aims to make tactical decisions to accomplish road following, obstacle avoidance, and efficient driving. In this work, we consider a typical but rarely studied task, called Target-Lane-Entering (TLE), where an autonomous vehicle should enter a target lane before reaching an intersection to ensure a smooth transition to another road. For navigation-assisted autonomous driving, a maneuver decision module chooses the optimal timing to enter the target lane in each road section, thus avoiding rerouting and reducing travel time. To achieve the TLE task, we propose a ruLe-aIded reiNforcement lEarning framework, called *LINE*, which combines the advantages of RL-based policy and rule-based strategy, allowing the autonomous vehicle to make target-oriented maneuver decisions. Specifically, an RL-based policy with a hybrid reward function is able to make safe, efficient, and comfortable decisions while considering the factors of target lanes. Then a strategy of rule revision aims to help the policy learn from intervention and block the risk of missing target lanes. Extensive experiments based on the SUMO simulator confirm the effectiveness of our framework. The results show that *LINE* achieves state-of-the-art driving performance with over 95% task success rate.

*Corresponding authors: Han Su and Kai Zheng.
†Kai Zheng is with Yangtze Delta Region Institute (Quzhou), School of Computer Science and Engineering, and Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China.

CCS CONCEPTS

• **Applied computing** → **Transportation**; • **Theory of computation** → *Reinforcement learning*.

KEYWORDS

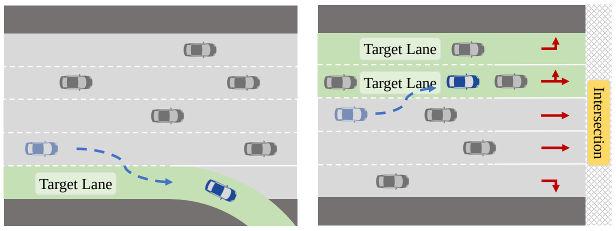
Autonomous Driving; Maneuver Decision; Reinforcement Learning

ACM Reference Format:
Ximu Zeng, Quanlin Yu, Shuncheng Liu, Yuyang Xia, Han Su, and Kai Zheng. 2023. Target-Oriented Maneuver Decision for Autonomous Vehicle: A Rule-Aided Reinforcement Learning Framework. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3583780.3615072

1 INTRODUCTION

Autonomous driving is a promising technology that has the potential to profoundly impact society by improving driving safety and enhancing traffic efficiency in intelligent transportation systems [38]. Maneuver decision, as the brain of an autonomous vehicle, determines the high-level driving commands in various traffic scenarios [37]. To achieve safe and efficient driving, it is desired to make proper decisions (lane and/or velocity changes) based on perceptual information, so that the subsequent controller can take fine-grained actions accordingly. The majority of the current maneuver decision research focuses on three autonomous driving tasks: road following, obstacle avoidance, and efficient driving [10, 16]. For road following, an autonomous vehicle follows a highway or arterial road safely by detecting the road markings and environments [25]. For obstacle avoidance, it maintains a safe distance from moving objects (vehicles or pedestrians) and stationary obstacles [39]. For efficient driving, it pursues a high velocity by overtaking the leading vehicles or changing to free lanes [13, 35].

Another task that is common in real life but rarely studied in literature is *Target-Lane-Entering (TLE)* task. Specifically, we consider that a vehicle follows the route from a navigation system, and it is required to *enter a target lane before arriving at an intersection* (crossroads, T-junction, roundabout, interchange, etc.), thus switching to another road smoothly. Human drivers mainly rely on



(a) Highway scenario (b) Urban scenario

Figure 1: Scenario Examples

the navigation system and environment to judge how to enter the proper lane, but they often miss the opportunity to change lanes in the following cases: 1) Human drivers may miss the lane change information that the navigation system mentions, which results in ignorance of entering a target lane. 2) When the vehicle is about to reach an intersection, human drivers may realize the requirement of entering certain target lanes, but the target lanes are not always available for inserting. Once the vehicle misses the target lanes and enters an unplanned road, it will deviate from the intended route and take longer travel time. Therefore, it is significant to help human drivers deal with the TLE task in daily driving.

Ideally, a well-designed maneuver decision module may solve the TLE task that human drivers are hard to tackle, and it should meet two potential needs of the task. Firstly, it should help an autonomous vehicle choose the proper timing to make lane changes that approach or enter the target lanes. Secondly, it needs to ensure that an autonomous vehicle is precisely in a target lane at the intersection, which conforms with the turning direction based on the planned route. The ideal module can better achieve autonomous driving in complex scenarios as examples in Figure 1. For one thing, an autonomous vehicle on highways or interchanges needs to enter the rightmost/leftmost lane when leaving the roads through off-ramps. For another, an autonomous vehicle on urban roads is required to turn or go straight at each intersection along a planned route. Since these scenarios frequently appear in any journey of end-to-end autonomous driving (traveling from an origin to a destination), the decision module should stand the TLE task.

The existing maneuver decision methods can be divided into rule-based and learning-based. Specifically, rule-based methods rely on predefined rules or heuristics to change lanes or velocity. Learning-based methods can be subdivided into two domains: Reinforcement Learning (RL) methods and Imitation Learning (IL) methods. RL-based methods train a policy to select maneuvers that maximize cumulative future rewards [16]. IL-based methods learn a policy by imitating human drivers' maneuvers [3, 5]. After investigating the above maneuver decision methods, we find that solely using these methods fails to achieve the TLE task for the following reasons. (1) Rule-based methods combine traditional decision-making models and default strategies to force autonomous vehicles to enter the target lane at the beginning or end of a road segment [15, 33]. Relying on rules tailored to specific driving scenarios, rule-based methods are strong in interpretability but are short in choosing tactical timing to change lanes. (2) RL-based methods have been proven to improve various driving metrics (e.g., efficiency and safety) by optimizing the cumulative reward [35, 39]. They can set task goals flexibly with well-designed reward functions, but they fail to ensure the successful completion of the TLE task due to decision uncertainty. (3) IL-based methods achieve autonomous driving by imitating expert maneuvers and can be used as prior knowledge for RL-based methods to reduce training time [17]. Although they are superior in the high utilization of empirical data, they struggle to collect expert demonstrations related to the TLE task. Overall, rule-based and RL-based approaches hold great potential to solve the TLE task, while IL-based methods are difficult to solve due to the lack of specific experts that adapt to the corner cases.

The above analyses motivate us to solve the TLE task by fusing rule-based and RL-based methods. An RL-based policy can choose the proper timing to change lanes, and a rule-based strategy can ensure task accomplishment. However, this intuition mainly faces three challenges. (1) The RL-based policy is required to comprehend how urgent the autonomous vehicle should enter target lanes. If the autonomous vehicle enters the target lane too early, it will encounter traffic jams when the density of the target lane is high. If the autonomous vehicle attempts to enter the target lane too late, it may hinder traffic flow or even cause accidents. (2) How to effectively fuse the RL-based policy and the rule-based strategy that achieves complementary advantages is complicated. The RL-based policy should maintain its intelligence while improving its reliability with the help of the rule-based strategy. (3) Since the TLE task is more complex than efficient driving, how to speed up the convergence efficiency of the RL-based policy needs to be studied.

To address these challenges, we propose a novel ruLe-aIded reiNforcement lEarning framework, named as *LINE*, to solve the TLE task. Given a meta scenario where an autonomous vehicle needs to travel from the upstream of the road to the downstream intersection and enter the target lanes, we first send the real-time perceptual state to an RL-based model with a hybrid reward function that takes the target lanes into consideration. We then employ rule-based guidance to help the RL-based policy learn from intervention and block the risk of missing target lanes. In order to enhance the training efficiency of the RL-based model under target lane conditions, we apply curriculum learning [2, 11, 23] to expose the model in increasingly complex tasks, that is, from a road following task to a safe interaction task and finally to the TLE task.

To sum up, the main contributions of this work are as follows:

- To the best of our knowledge, this is the first work to propose and address the target-lane-entering (TLE) problem that requires an autonomous vehicle to enter target lanes before arriving at an intersection.
- We develop a rule-aided reinforcement learning framework to deal with the TLE task, which achieves complementary advantages of RL-based policy and rule-based strategy.
- We propose a four-term hybrid reward function (safety, efficiency, comfort, and emergency) to enable the autonomous vehicle to make target-oriented maneuver decisions.
- We design a strategy of rule revision to supervise the maneuver decisions from the RL-based policy and also to guide the policy to learn from intervention.
- We conduct extensive experiments to evaluate the proposed framework in both meta and urban scenarios, verifying the effectiveness on both macroscopic and microscopic metrics.

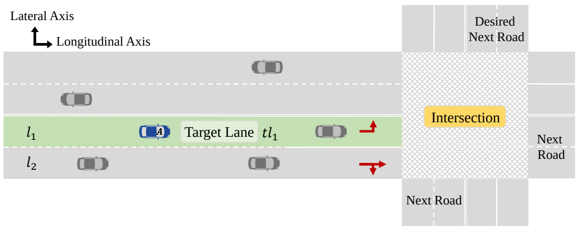


Figure 2: Example of Meta Scenario

2 PROBLEM STATEMENT

In this study, the maneuvers of an autonomous vehicle are controlled by the proposed model, *LINE*, in an interactive environment. In Figure 2, we present an example of the meta scenario. As shown, there is one autonomous vehicle A and a set of conventional vehicles C traveling on a multi-lane road ending at an intersection. The autonomous vehicle A can obtain real-time information about the locations and velocities of surrounding vehicles and itself through onboard sensors, such as planned navigation routes, Lidar, and IMU. This allows the autonomous vehicle to perform maneuvers (lane change action and velocity change action), at each time step t within a time duration $\mathbb{T} = \{1, 2, \dots, t, \dots\}$. We note that the crossroads is a representative scenario in intersections. Without loss of generality, we use the crossroads scenario as the meta scenario in the rest of this work.

Lane. A lane is part of a road to guide the traveling direction of vehicles. For example, lanes of a k-lane road are represented as $l_1, l_2, \dots, l_k$, and each lane has its allowed turning directions at the crossroads as shown in Figure 2. In order to follow the planned route, the autonomous vehicle A needs to enter a lane that conforms with the desired turning direction, which is defined as a *target lane*. Usually, there are multiple target lanes $\mathbb{TL} = \{tl_1, tl_2, \dots, tl_m\}$ among the k lanes, and the autonomous vehicle must be located in one of them before reaching the crossroads.

Location. A vehicle's location at time step t is denoted as a two-dimension position. The first axis points in the direction of a road, and the second axis is the direction perpendicular to it. For example, $(C_i^t.lat, C_i^t.lon)$ and $(A^t.lat, A^t.lon)$ represent the locations of C_i and A, respectively.

Velocity. The longitudinal velocities of C_i and A at time step t are denoted as $(C_i^t.v)$ and $(A^t.v)$, respectively. Considering the discrete time step, we assume the lateral motion between two consecutive time steps to be the uniform motion [18], and thus the lateral velocity change is ignored in this work. In the rest of this paper, velocity merely refers to longitudinal velocity.

Maneuver. At every time step t, the autonomous vehicle A performs a maneuver $(A^t.lc, A^t.vc)$ with two components. In details, $lc \in \{llc, rlc, lk\}$ is one of three lateral lane change action: left lane change action (llc), right lane change action (rlc) and lane keeping (lk). $vc \in [-acc, +acc]$ refers to the longitudinal velocity change action ranging from $-acc$ to acc.

Restriction. We pose some traffic restrictions on all vehicles.
(**1**) Speed limit. All vehicles are required to travel within the speed limit, i.e., v_{min} and v_{max}.
(**2**) Lane change restriction. Between two consecutive time steps, a vehicle can only make a lane change to an adjacent lane.
(**3**) Velocity change restriction. Vehicles are required to perform velocity changes that obey the acceleration restriction $[-acc, +acc]$.

Objective. In this work, our objective is that the autonomous vehicle can make safe, efficient, and comfortable decisions for entering a target lane before reaching an intersection.

3 METHODOLOGY

3.1 Framework Overview

Figure 3 shows the proposed rule-aided reinforcement learning framework that makes maneuver decisions as the autonomous vehicle interacts with the environment. It consists of three main components: environment modeling, RL-based policy, and rule revision.

Environment Modeling. We model the task as a Markov decision process and generate states that indicate the locations and velocities of the autonomous vehicle and surrounding vehicles, as well as the target lanes of the autonomous vehicle. To speed up the training process, we apply a curriculum learning strategy characterized by stages with increasing task complexity.

RL-based Policy. We pass the state as input of the RL-based policy module which then returns an action pair, including a discrete lane change action and a continuous velocity change action. Note that the reward is not sure yet, as the action pair may be revised via subsequent rule revision. As the example illustrated in Figure 3, the RL-based policy aims to maximize the reward where efficiency is stressed, and tends to make lane change at time step $t+2$ when the autonomous vehicle nearly arrives at the crossroads. However, it may fail to achieve the TLE task if the lane change action is blocked by other vehicles at time step $t+2$.

Rule Revision. To successfully complete the TLE task, we apply a strategy of rule revision into the decision-making process to revise the returned action generated by the RL-based policy. Specifically, rule-based guidance first compares the returned lane change action to a rule-based lane change action and then decides whether revision intervention is necessary based on two judgments. If intervention is needed, rule revision provides a revised action pair; otherwise, it outputs the returned action pair. According to the state before and after the action is executed, the reward is calculated using a four-term hybrid reward function.

3.2 Environment Modeling

Considering the mechanism by which an autonomous vehicle perceives its dynamic surrounding traffic and makes maneuver decisions, the TLE task conforms well to the principle of reinforcement learning. Based on the definitions of the TLE task, we model the maneuver decision of an autonomous vehicle, namely an agent A, as a Markov decision process (MDP) [1] with a hybrid discrete-continuous action space. Specifically, the MDP can be defined as: $\mathcal{M} =< \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma >$, which includes the state space $\mathcal{S}$, the discrete-continuous action space $\mathcal{A}$, the state transition probability distribution $\mathcal{P}$, the reward r, and the discount factor $\gamma \in [0, 1)$.

State. To help the agent A fulfill the TLE task, we extract a state matrix s^t from the environment at each time step t, including the state features of vehicles h_v^t and target lanes h_{tl}^t, i.e., $s^t = [h_v^t, h_{tl}^t]$.
(1) Vehicles. The vehicle features $h_v^t = [h_A^t, h_{c_1}^t, h_{c_2}^t, \dots, h_{c_6}^t]$ includes the feature vectors of the agent A and six surrounding

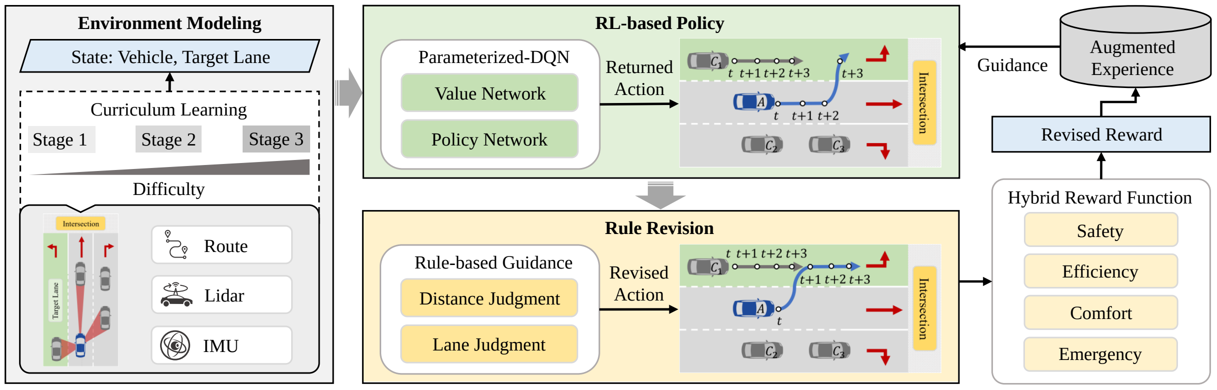


Figure 3: Framework Overview. Take the returned action and revised action in the figure as an example. Since the autonomous vehicle is about to reach the crossroads, the lane change action is revised from lk to llc at time step t to meet the TLE task.

conventional vehicles $C = \{C_1, C_2, \ldots, C_6\}$ [7]. To elaborate further, h_A^t includes the position and velocity features of agent A, denoted as $h_A^t = (A^t.lon, A^t.lat, A^t.v)$. Additionally, each $h_{c_i}^t$ characterizes the relative features between A and C_i, represented as $h_{c_i}^t = (d_{lon}(C_i^t, A^t), d_{lat}(C_i^t, A^t), v(C_i^t, A^t))$, where $d_{lon}(C_i^t, A)$ and $d_{lat}(C_i^t, A)$ refer to the relative longitudinal and lateral distances between A and C_i, respectively, and $v(C_i^t, A)$ denotes the relative longitudinal velocity. The relative features are calculated as follows:

$$h_{c_i}^t = (C_i^t.lon - A^t.lon, C_i^t.lat - A^t.lat, C_i^t.v - A^t.v) \tag{1}$$

(2) Target lanes. The features of target lanes h_{tl}^t on a k-lane road can be represented as $h_{tl}^t = (tgt_d^t, tgt_l^t)$, where tgt_d^t is a k-dimension vector to indicate the distribution of target lanes, and tgt_l^t is a 2-dimension vector serving as an auxiliary message to indicate turning direction. For example, consider a scenario where the agent travels on a 5-lane road before a crossroads, and its target lanes are the leftmost two lanes. In this case, tgt_d^t will be denoted as [1, 1, 0, 0, 0], where the codes of target lanes are set to 1 and the others are set to 0, and $tgt_l^t \in \{[0, 1], [1, 1], [1, 0]\}$ indicates the agent to turn right, keep straight, or turn left at the crossroads.

Action. Following the maneuver mentioned in Section 2, the action of the agent A can be regarded as a hybrid discrete-continuous action pair, which can be represented as follows:

$$a^t = (A^t.lc, A^t.vc), \tag{2}$$

where $A^t.lc \in \{llc, rlc, lk\}$ is the lane change action with discrete action space, and $A^t.vc \in [-acc, +acc]$ is the velocity change action with continuous action space.

State Transition. After the agent performs an action a^t, we update the state s^t into the next state s^{t+1}. For vehicle features, the feature vector of agent $h_A^{t+1} = (A^{t+1}.lon, A^{t+1}.lat, A^{t+1}.v)$ at time step $t+1$ is updated as follows:

$$\begin{aligned} A^{t+1}.lon &= A^t.lon + \Delta t \cdot A^t.v + \frac{1}{2} A^t.vc \cdot (\Delta t)^2 \\ A^{t+1}.lat &= A^t.lat + \overline{A^t.lc} \cdot L_w \\ A^{t+1}.v &= A^t.v + \Delta t \cdot A^t.vc \end{aligned} \tag{3}$$

where Δt denotes the time interval between two consecutive time steps, L_w is the width of a lane, and $\overline{A^t.lc}$ is equal to −1, 0, and +1 in the case that lane change action is llc, lk, and rlc, respectively. The feature vector $h_{c_i}^{t+1}$ of each conventional vehicle $C_i \in C$ that is controlled by a human-like algorithm, is then obtained via on-board sensors. For target lane features, we update the codes in h_{tl}^t according to the planned route.

Reward. The agent A receives a reward r^t according to the action a^t and the state transition from s^t to s^{t+1}. After testing various reward combinations, we design a four-term reward function that takes 1) safety, 2) efficiency, 3) comfort, and 4) emergency into account. The reward function is introduced in Section 3.3 in detail.

Curriculum Learning. Given the difficulty of the TLE task compared to simpler tasks like road following and obstacle avoidance, we apply curriculum learning to speed up the learning process. When human beings learn how to drive a car, the learning process is a natural curriculum that starts with speed control and gradually transits to both speed and steering control. Inspired by human nature in learning, we adopt the strategy of curriculum learning and break the whole training process into three stages with growing complexity. By training in a process with specialized stages, the agent A can effectively fine-tune network parameters learned from earlier stages to adapt to the subsequent stages [2, 11].

(1) Stage 1: road following. To facilitate the learning of efficient driving with less jerk, the agent A is required to simply drive along a road in low traffic density. Two measures are employed to help the agent A focus on its velocity change. First, when the agent A is generated at the starting point, it is positioned far away from other vehicles in the low-density environment. This spatial arrangement minimizes the attention for agent A to extensive interaction with other vehicles. Second, to encourage the agent A to prioritize smooth driving, the reward value of emergency is intentionally not stressed in Stage 1 with no target lane.

(2) Stage 2: safe interaction. To learn a safer driving strategy in more congested traffic conditions compared to Stage 1, the agent A needs to drive along a road in high traffic density without target lanes. Since more vehicles are likely to travel around, the agent A should maintain a safe distance from its surrounding vehicles, which aligns with the reward value of safety.

(3) Stage 3: the TLE task. Based on Stage 2, Stage 3 further incorporates the target lanes. Even though the agent A can perform change

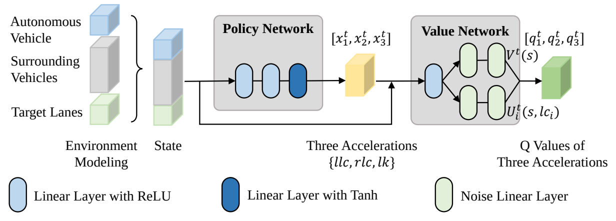


Figure 4: Network Structure in *LINE*

lanes for efficiency and safety in Stages 1 and 2, lane change action that considers target lanes is more demanding in Stage 3.

3.3 RL-based Policy for Maneuver Decision

Based on the environment modeling, we design a reinforcement learning-based model to compute a discrete-continuous action of the autonomous vehicle. The model consists of three key components: optimization paradigm, network structure, and reward function. Notably, the reward value of emergency is carefully designed to make target-oriented maneuver decisions.

Optimization. Given a state s^t, the goal of the agent A is to learn a policy u with maximal expected γ-discounted cumulative reward, i.e., action-value function Q^* [22], as follows:

$$Q^*(s^t, a^t) = \max_u \mathbb{E}\left[\sum_{t' \geq t}^{T} \gamma^{t'-t} r(s^{t'}, a^{t'})|u\right] \tag{4}$$

where $\gamma \in [0, 1)$ is a discount factor. Based on the discrete-continuous action space of our TLE task, we apply P-DQN [36] as the reinforcement learning paradigm, which defines a Bellman equation [21, 27] to estimate Q^* as follows:

$$\begin{aligned} Q(s^t, a^t) = Q(s^t, A^t.lc, A^t.vc) = \mathbb{E}_{s^{t+1}}\Big[r(s^t, a^t)+ \\ \gamma \max A^{t+1}.lc \in \{llc, rlc, lk\} Q(s^{t+1}, A^{t+1}.lc, u(s^{t+1}, A^{t+1}.lc))\Big] \end{aligned} \tag{5}$$

where the *max* operation gets an optimal lane change action from $\{llc, rlc, lk\}$ and policy u gets an optimal velocity change action.

Network Structure. As shown in Figure 4, the network structure is divided into a policy network and a value network. The policy network outputs the accelerations of each discrete lane change action, and the value network outputs Q values of each acceleration.

(1) Policy network. The input is the state $s^t = [h_v^t, h_{tl}^t]$, and the output $X^t = [x_1^t, x_2^t, x_3^t]$ are three velocity change actions of each lane change action $lc_i \in \{A^t.lc = llc, A^t.lc = rlc, A^t.lc = lk\}$, i.e., $x_i^t = A^t.vc|_{lc_i}$. The output X^t is calculated as follows:

$$X^t = acc \cdot Tanh(\phi_3(ReLU(\phi_2 ReLU(\phi_1 s^t + b_1) + b_2) + b_3)) \tag{6}$$

where acc is the acceleration bound, ϕ_1, ϕ_2 and ϕ_3 are linear transformations, and b_1, b_2 and b_3 are biases. Since Tanh function can limit output in $[-1, 1]$ as an activation function, the values of velocity change actions in X^t are limited in $[-acc, +acc]$.

(2) Value network. It receives both the state s^t and the X^t as input and processes them through a linear transformation at first. Similar to Rainbow DQN [12], the dueling technique [32] is used later to estimate Q values as the output $Q^t = [q_1^t, q_2^t, q_3^t]$. In detail, the dueling technique combines a value stream and an advantage stream that are both composed of noise linear layers [26]. The value stream estimates the state value $V^t(s)$ of the general state, while the advantage stream estimates the relative state-action value $U_i^t(s, lc_i)$ of each lane change action. The state value $V^t(s)$ and the relative state-action value $U_i^t(s, lc_i)$ are calculated as follows:

$$\begin{aligned} V^t(s) &= \phi_6(\phi_5(ReLU(\phi_4 X^t + b_4))) \\ U_i^t(s, lc_i) &= \phi_8(\phi_7(ReLU(\phi_4 X^t + b_4))) \end{aligned} \tag{7}$$

where ϕ_4 and b_4 are a linear transformation and its bias, ϕ_5, ϕ_6, ϕ_7 and ϕ_8 denote noise linear transformations.

The output Q^t is then calculated as follows:

$$q_i^t = V^t(s) + U_i^t(s, lc_i) - \frac{1}{|U|}\sum_{i=1}^{|U|} U_i^t(s, lc_i) \tag{8}$$

where $|U| = 3$ is the number of discrete actions in action space.

Hence, the policy u selects the lane change action $A^t.lc$ with the highest Q values, and its corresponding continuous velocity change actions $A^t.vc$ as follows:

$$\begin{aligned} A^t.lc &= \arg\max_{x_i^t \in X^t} Q_{out}^t \\ A^t.vc &= x_i^t|_{A^t.lc} \end{aligned} \tag{9}$$

The action $a^t = (A^t.lc, A^t.vc)$ is then revised via rule-based guidance, with detailed design in Section 3.4.

Reward Function. The reward function is a highly important role to guide the optimization of a policy. To deal with the TLE task, we construct a hybrid reward function concerning four elements, 1) safety, 2) efficiency, 3) comfort, and 4) emergency. The reward at time step t is defined as follows:

$$r^t = w_1 r_s^t + w_2 r_e^t + w_3 r_c^t + w_4 r_m^t \tag{10}$$

where w_1, w_2, w_3, w_4 are four weights to represent the importance of each part, and $r_s^t, r_e^t, r_c^t, r_m^t$ refer to the reward values of safety, efficiency, comfort, and emergency, respectively.

(1) Safety. Time to collision (TTC) [9] is used to indicate the safety of maneuver decisions generated by the agent A. Specifically, TTC is the remaining time before two vehicles collide, supposing no alteration of speed or lane during the remaining time. Since we do not focus on lateral velocity, lateral TTC is not considered. The TTC^t between the agent A and its leading conventional vehicle C_i is valid when $v(C_i^t, A^t) < 0$, which can be defined as:

$$TTC^t = \begin{cases} -\frac{d_{lon}(C_i^t, A^t)}{v(C_i^t, A^t)}, & v(C_i^t, A^t) < 0 \\ 0, & v(C_i^t, A^t) \geq 0 \end{cases} \tag{11}$$

With TTC^t, the reward value of safety $r_s^t \in [-10, 0]$ is a term penalizing dangerous driving behaviors, which is defined as:

$$r_s^t = \begin{cases} -10, & collision \\ max(-2, log(\frac{TTC^t}{\delta})), & 0 < TTC^t \leq \delta \\ 0, & otherwise \end{cases} \tag{12}$$

where $collision$ infers the agent A encountering collision, and δ denotes a δ-second time limitation for TTC.

(2) Efficiency. The reward value of efficiency [18] $r_e^t \in [0, 1]$ is a term encouraging the agent A to move forward efficiently within

the speed limit, which is defined as follows:

$$r_e^t = \begin{cases} 0, & A_v^t > v_{max} \\ \frac{A_v^t}{v_{max}}, & 0 \le A_v^t \le v_{max} \end{cases} \tag{13}$$

where v_{max} is the speed upper limit. Note that the autonomous vehicle is able to speeding without a clip for A_v^t, but such behavior will lead to a zero reward value of r_e.

(3) Comfort. As a penalty term discouraging a high change rate of velocity that brings a negative jerk experience for passengers [39], the reward value of comfort $r_c^t \in [-1, 0]$ is defined as:

$$r_c^t = -\frac{|A^t.vc - A^{t-1}.vc|^2}{(acc - (-acc))^2} \tag{14}$$

(4) Emergency. We introduce a term r_m^t in the reward function that reflects the emergency associated with changing lanes to enter the target lanes for two reasons. First, even if the target lanes are constructed in state s^t, the policy u can hardly notice target lanes during optimization without the feedback of target lanes in the reward function. Second, an intuitive way to design the reward value for target lanes is to measure the number of lanes from a target lane. However, such a design may force the agent A to move to a target lane too early which potentially leads to low efficiency, because the urgency and difficulty of achieving the TLE task varies in different states. Hence, we design an emergency penalty r_m^t that increases when the agent A leaves further away from target lanes $\mathbb{TL}$ and/or gets closer to the crossroads, which is calculated as:

$$r_m^t = -\frac{A^t.lon}{L_l} \cdot \frac{min(|A^t.lat - tl_i.c|)}{L_w \cdot L_n} \tag{15}$$

where $tl_i.c$ is the lateral center position of a target lane $tl_i \in \mathbb{TL}$, L_l is the total length of a road, and L_n is the number of lanes on a road. Since all target lanes can be regarded as the same for the TLE task, we use the *min* function to choose the closest distance between the agent A and target lanes $\mathbb{TL}$.

3.4 Rule Revision

We provide a rule-based strategy to help target-oriented maneuver decisions for two motivations. (1) Task guarantee. Learning-based methods alone (e.g., reinforcement learning and imitation learning) are insufficient to ensure the fulfillment of the TLE task [14, 20, 30], while rule revision can serve as a guaranteed measure to meet the TLE task by altering the maneuver decision of RL-based policy. However, a challenge lies in determining when to intervene with rule-based guidance in different situations. To deal with this challenge, we use a pipeline in Algorithm 1 to determine whether to use rule-based guidance in a certain state. (2) Optimization feedback. Apart from monitoring and modifying the agent's maneuver decisions, rule revision can also guide the agent to learn from intervention via experience augmentation during the learning progress [34]. In detail, besides normal state transitions, extra state transitions that are intervened by rule-based guidance are also saved in experience.

Rule-based Guidance. As depicted in Algorithm 1, a pipeline that considers both distance and lane judgment can revise the action pair before execution at every time step. The pipeline receives the state s^t and an action pair $a^t = (A^t.lc, A^t.vc)$ generated by the maneuver decision of RL-based policy, and then outputs the revised action

Algorithm 1: Intervention of rule-based guidance

```
Input: state $s^t$, returned action $a^t = (A^t.lc, A^t.vc)$,
       $\mathcal{R}^t.lc \in \{llc, rlc, lk\}$
Output: revised action $a^{t\prime} = (A^{t\prime}.lc, A^{t\prime}.vc)$
1  $a^{t\prime}$ is set as $a^t$;
2  initialize no intervention;
3  if $A^t.lc \ne \mathcal{R}^t.lc$ and $A$ is far away from the crossroads then
4      // not suitable to leave a target lane
5      if $A$ in $\mathbb{TL}$ then
6          implement intervention;
7      else
8          no intervention;
9  // urgent need to act as $\mathcal{R}^t.lc$
10 if $A^t.lc \ne \mathcal{R}^t.lc$ and $A$ is about to reach the crossroads then
11     implement intervention;
12 if implement intervention then
13     $A^{t\prime}.lc$ is set as $\mathcal{R}^t.lc$;
14     $A^{t\prime}.vc$ is set as $A^t.vc|_{A^t.lc=\mathcal{R}^t.lc}$;
```

pair $a^{t\prime} = (A^{t\prime}.lc, A^{t\prime}.vc)$ with or without intervention. To indicate the mechanism of the rule, we introduce $\mathcal{R}^t.lc \in \{llc, rlc, lk\}$ to show a rule-based lane change action approaching the nearest target lane, i.e., $\mathcal{R}^t.lc = lk$ when the agent is on a target lane. Moreover, two judgments decide the necessity of intervention. A distance judgment is based on the distance from the crossroads, and a lane judgment is based on the target lanes. Specifically, the pipeline first compares the returned lane change action $A^t.lc$ with the rule-based lane change action $\mathcal{R}^t.lc$. Two types of intervention are triggered when the actions are inconsistent. 1) When the agent is far away from the crossroads and travels exactly on a target lane, the rule-based guidance makes the agent keep moving straight in the target lane as $\mathcal{R}^t.lc = lk$ (lines 3-8). 2) When the agent is about to reach the crossroads ahead, it is advised to move to a target lane promptly (lines 9-11). Hence, the rule-based guidance makes the agent act as $\mathcal{R}^t.lc$. When intervention is implemented in the above two situations, the pipeline then sets the revised action pair $A^{t\prime}.lc$ as $\mathcal{R}^t.lc$ and sets the revised velocity change action $A^{t\prime}.vc$ correspondingly (lines 12-14). Otherwise, the revised action pair $a^{t\prime}$ is the same as the return action pair a^t.

Reward Revision. As a measure of experience augmentation, the state transition of returned action a^t with reward r^t and revised action $a^{t\prime}$ with reward $r^{t\prime}$ are both saved in the experience memory of RL-based policy. To indicate the deficiency of a^t when intervention happens, its reward r^t is further revised as follows:

$$\begin{aligned} r^t &= r^{t\prime} + \rho \\ \rho &= w_5|A^{t\prime}.lc - A^t.lc| + w_6|A^{t\prime}.vc - A^t.vc| \end{aligned} \tag{16}$$

where ρ describes the difference between $a^t = (A^t.lc, A^t.vc)$ and revised action $a^{t\prime} = (A^{t\prime}.lc, A^{t\prime}.vc)$, w_5 and w_6 are combination weights. It is notable that in the following experiments, rule-based guidance and reward revision are only applied in the training process, rather than in both the training and testing process. This means that we try to use rule revision to guide the RL policy to learn from intervention and to optimize the networks correspondingly.

4 EXPERIMENTS

To verify the effectiveness of our framework, we conduct comprehensive experiments of *LINE* from both macroscopic and microscopic aspects. We first introduce experiments in meta scenarios, while those for urban scenarios are illustrated in Section 4.6.

4.1 Experimental Settings

Implementation Details. To simulate the interaction between the autonomous vehicle and its surrounding conventional vehicles, all experiments are conducted in an open-source simulation environment, SUMO [19]. Without loss of generality, we simulate a straight five-lane road in meta scenarios, and a grid road network in urban scenarios, where a road segment spans $2km$ and the width of each lane L_w is $3.2m$. To reflect real-world traffic constraints and to follow previous works [18, 35], we set traffic limitations as $v_{min} = 5km/h \approx 1.39m/s$, $v_{max} = 90km/h = 25m/s$, and $acc = 3m/s^2$. The detection range of sensors on the autonomous vehicle is set to $100m$. The time granularity between consecutive time steps is $\Delta t = 0.5s$ [37]. The time threshold in TTC is set as $\delta = 4$. The different traffic densities in curriculum stages are set as, 200 vehicles per kilometer in high-density stages and 100 vehicles per kilometer in low-density stages.

We conduct 4000 episodes for the training process and 500 for the testing process. An episode involves the autonomous vehicle driving from the origin to the crossroads ahead in meta scenarios, to the destination in urban scenarios, or to the location of a collision before its arrival. Two measures are implemented to mimic the real traffic patterns so that the agent can be trained in diverse episodes. First, all vehicles are initialized in random lanes, and the departure sequence of the autonomous vehicle among the fleet is randomized. Second, conventional vehicles also vary in driving habits like the aggression of lane changes.

The Adam optimizer is used to train the network with a batch size of 128 and a learning rate of 0.001. Following P-DQN [36], target networks are added to update the network in *LINE* with a soft update ratio of 0.01. Via grid searching, the coefficients in the reward function are set as $w_1 = 1.0$, $w_2 = 0.4$, $w_3 = 1.0$, and $w_4 = 2.0$, and the combination factors in the reward revision are set as $w_5 = 0.5$ and $w_6 = 1$. The epsilon greedy strategy is set as 0.05 in exploration.

Baselines. Since nearly no existing work can solve the TLE task, we add the rule revision or the hybrid reward function in a few representatives as baselines. The detailed baselines are as follows:
(1) IDM-LC [15]. Traditional intelligent driver model with a lane-changing model used in SUMO [19] for maneuver decision.
(2) ILR. An imitation learning method that tries to learn a rule-based maneuver decision strategy with expert demonstrations in IDM-LC.
(3) D3QN-Reward [32]. A classical DRL model that outputs discrete lane change and velocity change actions with the same reward function in *LINE*.
(4) PDQN-Rule [36]. A DRL model that uses a vanilla P-DQN with the same rule revision strategy proposed in *LINE*.

4.2 Macroscopic Evaluation

We use macroscopic metrics to illustrate the intuitive performance of a whole episode. Four metrics (i.e., Suc, Col, AvgRR, AvgLC) that are related to the fulfillment of the TLE task are introduced at first. We then present a metric about traveling time (AvgT) and another metric about affection (AvgAff). The macroscopic metrics are designed as follows:
(1) Success rate. We record the success rate of task accomplishment of the autonomous vehicle (Suc). We regard task accomplishment to be successful in an episode when the autonomous vehicle reaches the crossroads on a target lane. The larger value of Suc implies that the autonomous vehicle learns the task better.
(2) Collision rate of colliding with conventional vehicles (Col). We record the possibility of an autonomous vehicle colliding with conventional vehicles in an episode. The smaller value of Col implies higher safety for the autonomous vehicle.
(3) Average number of rule revisions (AvgRR). The smaller value of AvgRV implies that the autonomous vehicle has a good policy for maneuver decisions with fewer rule interventions. AvgRR is not considered when rule revision is not applied in a method.
(4) Average number of lane changes for the autonomous vehicle from the origin to the crossroads (AvgLC). The smaller value of AvgLC implies that the autonomous vehicle takes a more rigorous lane change strategy.
(5) Average traveling time of the autonomous vehicle traveling from the origin to the crossroads (AvgT). The smaller value of AvgT implies that the autonomous vehicle has better driving efficiency.
(6) Average affection time that the autonomous vehicle has a negative impact on its following conventional vehicle (AvgAff) [18, 35]. The smaller value of AvgAff implies a less negative impact on the following vehicle.

Table 1: Macroscopic Performance of Baselines and *LINE*

Method	Suc (%)	Col (%)	Avg-RR	Avg-LC	AvgT (s)	Avg-Aff
IDM-LC	98.6	**0**	-	1.95	98.91	13.93
ILR	85.2	10.6	-	6.82	92.41	12.84
D3QN-Reward	89.4	8.4	-	7.49	**81.65**	11.52
PDQN-Rule	94.0	5.6	0.758	4.95	87.57	10.90
LINE	**99.2**	**0**	**0.442**	**1.53**	84.36	**10.22**

Table 2: Microscopic Performance of Baselines and *LINE*

Method	MinTTC (s)	AvgV (m/s)	AvgJ (m/s^2)
IDM-LC	2.60	20.22	0.2725
ILR	2.73	21.64	0.2892
D3QN-Reward	1.13	**24.49**	0.2355
PDQN-Rule	3.04	22.83	0.2224
LINE	**3.35**	23.70	**0.2161**

Results Analysis. Table 1 reveals the macroscopic performance of *LINE* compared with baselines. Regarding the fulfillment of the TLE task, *LINE* achieves the highest success rate (Suc) with the fewest rule revisions (AvgRR), the most optimal lane change decisions (AvgLC), and nearly no collision happening (Col). In addition, *LINE* performs efficiently with a low AvgT, resulting in a reduction of 3.6%-14.7% in travel time. The reason lies in that *LINE* enables the autonomous vehicle to travel with high efficiency via the maneuver decision based on the surrounding traffic information provided in the input state. Even though no term directly indicates affection to the following vehicle (AvgAff) in the reward function, we evaluate

Table 3: Macroscopic and Microscopic Performance of Variants and *LINE*

Method	Macroscopic						Microscopic		
	Suc (%)	Col (%)	AvgRR	AvgLC	AvgT (s)	AvgAff	MinTTC (s)	AvgV (m/s)	AvgJ (m/s^2)
LINE-w/o-CL	96.4	2.8	0.468	2.02	86.94	10.70	2.42	23.00	0.2648
LINE-w/o-RV	91.8	4.2	-	1.76	86.17	10.43	3.20	23.21	0.2234
LINE-w/o-emg	95.2	3.6	0.896	1.75	89.08	10.40	2.71	22.45	0.2195
LINE	**99.2**	**0**	**0.442**	**1.53**	**84.36**	**10.22**	**3.35**	**23.70**	**0.2161**

it because hash deceleration and casual lane change behaviors can cause AvgAff indirectly, which are related to the reward value of comfort and emergency, respectively. *LINE* also shows superior performance with the lowest AvgAff which saves 6.2%-26.6% affection time. This can be explained that *LINE* effectively avoids unnecessary lane changes with the target lanes considered.

Among the other methods, IDM-LC is another method that achieves high Suc, but it falls short in achieving efficient driving with the longest AvgT. ILR, which attempts to mimic the policy in IDM-LC, performs no better than IDM-LC generally. Lacking rule revision or the ability for continuous velocity change, D3QN-Reward is more likely to collide with other vehicles (Col). Even though D3QN-Reward spends the least AvgT, it has the worst performance in terms of Suc and AvgLC, for it may act in a hasty way. Owning to the aid of rule revision, PDQN-Rule performs better than ILR and D3QN-Reward. However, PDQN-Rule requires more rule intervention (AvgRR), since its maneuver decision policy is weaker than that of *LINE*.

4.3 Microscopic Evaluation

We use microscopic metrics to illustrate the hint performance of a time step. The microscopic metrics are designed as follows:
(1) Minimum TTC of the autonomous vehicle (MinTTC). We record the minimum TTC between the autonomous vehicle and its leading vehicle. The smaller value of MinTTC implies that the autonomous vehicle tends to remain an unsafe distance from its leading vehicle.
(2) Average velocity of the autonomous vehicle (AvgV). We record the velocity of the autonomous vehicle at every time step. The larger value of AvgV implies that the autonomous vehicle stays at high efficiency.
(3) Average jerk of the autonomous vehicle (AvgJ). We also record the jerk value of the autonomous vehicle at every time step. The smaller value of AvgJ implies that the autonomous vehicle has fewer reckless velocity changes.
Results Analysis. As illustrated in Table 2, *LINE* achieves the highest MinTTC, a high AvgV, and the lowest AvgJ. This indicates that *LINE* can keep a safe following distance and stay at a stable high speed simultaneously. Due to the intrinsic limitation in rules, IDM-LC tends to prioritize safety over efficiency, resulting in the lowest average velocity (AvgV). ILR still performs like IDM-LC but is no better than IDM-LC, because it struggles to fully understand the TLE task and fails to capture the complex determination process in IDM-LC. Consistent with the macroscopic evaluation, D3QN-Reward tends to travel impulsively with the lowest MinTTC and the highest AvgV. Different from *LINE* in network structure, reward function, and curriculum learning strategy, the performance of PDQN-Rule is not as good as that of *LINE*.

4.4 Ablation Study

We conduct an ablation study to evaluate the effectiveness of some key components in *LINE* when tackling the TLE task. Both the macroscopic and microscopic metrics in Section 4.2 and 4.3 are applied in the ablation study.
Variants. We study three variants that are related to curriculum learning, rule revision, and emergency reward, respectively. The detailed set of variants is as follows:
(1) *LINE*-w/o-CL. It is trained in Stage 3 during the whole training process without the strategy of curriculum learning.
(2) *LINE*-w/o-RR. The executed maneuver is exactly what the agent outputs without rule revision.
(3) *LINE*-w/o-emg. The emergency is removed from the reward function, while curriculum learning and rule revision still work.
Results Analysis. From Table 3, it can be concluded that *LINE* shows superior performance than all variants. Compared with *LINE*-w/o-CL, *LINE* with curriculum learning demonstrates a more advanced maneuver decision strategy, especially in terms of success rate (Suc) and collision rate (Col). The reason is that policy with curriculum learning can avoid overfitting with noises generated from different training stages, and thus leads to a better performance in the TLE task. Compared with *LINE*-w/o-RR, *LINE* with rule revision achieves a higher success rate (Suc). This can support the advantage that both RL and rule-guidance experience are replayed when experience is used during policy optimization. Compared with *LINE*-w/o-emg, *LINE* with the reward value of emergency exhibits a lower average number of rule revisions (AvgRR). The reason lies in that this reward component, emergency, also plays an important role when the reward function works, so that the policy itself is able to fulfill the task without much aid from rule revision.

4.5 Training Efficiency

To illustrate the effectiveness of curriculum learning that is expected to speed up the training process, we compare the training efficiency of *LINE* and several baselines and variants (i.e., D3QN-Reward, PDQN-Rule, *LINE*-w/o-CL). Visualized as learning curves in Figure 5, solid lines represent the mean values of average reward and shaded areas imply the standard deviations during the training process. The stages of curriculum learning applied in *LINE* are distinguished with green dashed lines.
Results Analysis. As we can see in Figure 5, the learning curve of *LINE* stands out with its remarkable stability and superior performance compared to the other curves. Benefiting from training in a simpler stage at first, *LINE* begins with the highest average reward in the early episodes. The strategy of curriculum learning also helps *LINE* to steadily converge in each stage and to rapidly achieve the TLE task, as we can see that the curve of *LINE* shows a

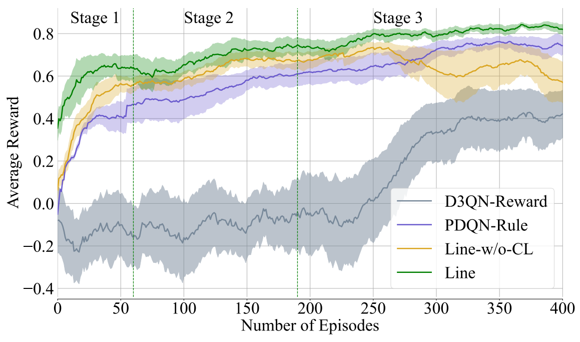


Figure 5: Effectiveness of Curriculum Learning

downward trend when transitioning to a new stage. On the contrary, the learning curves of PDQN-Rule and *LINE*-w/o-CL that lack the implementation of curriculum learning exhibit fluctuations and a downward trend in the later episodes. As for D3QN-Reward, it starts and ends with the lowest average reward for its inherent limitation of coarse granularity in discrete velocity change actions.

4.6 End-to-end Extension

We further conduct an experiment to compare *LINE* and baselines in a more complex urban scenario, which extends a multi-lane road to an end-to-end route in a grid road network.

Extension Experiment Settings. Similar to the objective in the former experiments, in the end-to-end extension experiment, the autonomous vehicle is required to make safe and efficient maneuver decisions while traveling from a start point to the destination along a planned route. Precisely, the entire route consists of five two-kilometer-long roads, and the start point and destination are randomly selected in the grid road network. To successfully travel along the route, the autonomous vehicle needs to achieve the TLE task sequentially when traveling on different road segments with corresponding target lanes, while it is controlled by classical rules [28] when traveling through a crossroads. We use macroscopic metrics in the extension experiment to evaluate the performance in the whole route, since microscopic metrics are not measurable in crossroads.

Results Analysis. As shown in Table 4, the disparities between *LINE* and baselines on the success rate (Suc) and the collision rate (Col) become significant in the end-to-end urban scenario. This can be explained that the fulfillment of the TLE task on one road segment affects the completion of the TLE task on the whole route, and the overall success rate of the entire route is influenced by the cumulative success rates of individual road segments. With more TLE tasks to tackle, *LINE* outperforms in all metrics except for AvgT. The relative performance between methods is consistent with what we analyze in experiments of meta scenarios.

5 RELATED WORK

Rule-based Methods. Rule-based methods are often used in low-level control tasks, such as lane following and speed control. A classical control algorithm in autonomous driving is the Proportional-Integral-Derivative (PID) controller [24], which calculates the steering angle to keep the vehicle on the center line of a lane. Adaptive Cruise Control (ACC) [29, 33] is a cruise control system that helps ego vehicle to adjust speed maintenance and safe following distance, via detecting the speed and location of the leading vehicle with sensors such as radar or lidar.

Table 4: Performance of Baselines and *LINE* in Extension Experiments

Method	Suc (%)	Col (%)	Avg-RR	Avg-LC	AvgT (*s*)	Avg-Aff
IDM-LC	92.4	**0**	-	10.47	513.87	74.53
ILR	43.6	43.8	-	36.04	506.84	69.27
D3QN-Reward	56.8	37.2	-	42.45	**416.31**	58.62
PDQN-Rule	72.6	20.4	3.816	21.39	458.29	55.31
LINE	**95.6**	**0**	**2.267**	**7.93**	435.35	**52.60**

RL-based Methods. Reinforcement learning can be used to train an agent to interact with the driving environment based on input from sensors. [31] defines action space as vehicle yaw acceleration and designs a Q-function approximator with a closed-form greedy policy in DQN. [4] breaks down overall behavior into sub-policies with a hierarchical action structure that fits lane change behavior. Later, more works are proposed to solve more problems in autonomous driving, such as dealing with harsh velocity changes [35] and using space-weighted information fusion [8].

IL-based Methods. Imitation learning involves learning a policy that leverages human expertise and driving skills, allowing the autonomous vehicle to imitate the behavior of expert drivers. To improve the safety of maneuver decisions using IL, [3] proposes a framework that can handle complex urban scenarios with IL and an enhanced safety controller. [6] trains an IL agent with an attention model to receive image input and output steering angles.

Rule-based methods struggle to handle various scenarios flexibly due to the reliance on predefined rules. RL-based methods and IL-based methods learn from the trial-and-error process and expert demonstrations, respectively, but they can neither guarantee the successful completion of the TLE task. Hence, one shared limitation among these three types of methods is their inability to solely address the TLE task.

6 CONCLUSION

In this work, we propose a novel rule-aided reinforcement learning framework, named *LINE*, to address the TLE task in autonomous driving. *LINE* first uses an RL-based policy with a four-term hybrid reward function to make target-oriented maneuver decisions. Then *LINE* applies rule revision with distance and lane judgment to supervise and guide the RL-based policy. Through experiments in both meta and urban scenarios, we demonstrate the superiority of *LINE* over state-of-the-art methods in the TLE task with macroscopic and microscopic metrics.

ACKNOWLEDGMENTS

This work is partially supported by NSFC (No. 61972069, 61836007, 61832017, 62272086), Shenzhen Municipal Science and Technology R&D Funding Basic Research Program (JCYJ20210324133607021), Municipal Government of Quzhou under Grant No. 2022D037, and Key Laboratory of Data Intelligence and Cognitive Computing, Longhua District, Shenzhen.

REFERENCES

[1] Szilárd Aradi. 2020. Survey of deep reinforcement learning for motion planning of autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems* 23, 2 (2020), 740–759.

[2] Jin Chen, Guanyu Ye, Yan Zhao, Shuncheng Liu, Liwei Deng, Xu Chen, Rui Zhou, and Kai Zheng. 2022. Efficient Join Order Selection Learning with Graph-based Representation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 97–107.

[3] Jianyu Chen, Bodi Yuan, and Masayoshi Tomizuka. 2019. Deep imitation learning for autonomous driving in generic urban scenarios with enhanced safety. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2884–2890.

[4] Yilun Chen, Chiyu Dong, Praveen Palanisamy, Priyantha Mudalige, Katharina Muelling, and John M Dolan. 2019. Attention-based hierarchical deep reinforcement learning for lane change behaviors in autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 0–0.

[5] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. 2022. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

[6] Luca Cultrera, Lorenzo Seidenari, Federico Becattini, Pietro Pala, and Alberto Del Bimbo. 2020. Explaining autonomous driving by learning end-to-end visual attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 340–341.

[7] Shengzhe Dai, Li Li, and Zhiheng Li. 2019. Modeling vehicle interactions via modified LSTM models for trajectory prediction. *IEEE Access* 7 (2019), 38287–38296.

[8] Jiqian Dong, Sikai Chen, Yujie Li, Runjia Du, Aaron Steinfeld, and Samuel Labi. 2021. Space-weighted information fusion using deep reinforcement learning: The context of tactical control of lane-changing autonomous vehicles and connectivity range assessment. *Transportation Research Part C: Emerging Technologies* 128 (2021), 103192.

[9] Leonard Evans. 1991. *Traffic safety and the driver.* Science Serving Society.

[10] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. 2020. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics* 37, 3 (2020), 362–386.

[11] Guy Hacohen and Daphna Weinshall. 2019. On the power of curriculum learning in training deep networks. In *International Conference on Machine Learning*. PMLR, 2535–2544.

[12] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. 2018. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[13] Carl-Johan Hoel, Katherine Driggs-Campbell, Krister Wolff, Leo Laine, and Mykel J Kochenderfer. 2019. Combining planning and deep reinforcement learning in tactical decision making for autonomous driving. *IEEE transactions on intelligent vehicles* 5, 2 (2019), 294–305.

[14] Zhiyu Huang, Jingda Wu, and Chen Lv. 2022. Efficient deep reinforcement learning with imitative expert priors for autonomous driving. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[15] Arne Kesting, Martin Treiber, and Dirk Helbing. 2010. Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 368, 1928 (2010), 4585–4605.

[16] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. 2021. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems* 23, 6 (2021), 4909–4926.

[17] Luc Le Mero, Dewei Yi, Mehrdad Dianati, and Alexandros Mouzakitis. 2022. A survey on imitation learning techniques for end-to-end autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems* (2022).

[18] Shuncheng Liu, Han Su, Yan Zhao, Kai Zeng, and Kai Zheng. 2021. Lane change scheduling for autonomous vehicle: A prediction-and-search framework. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 3343–3353.

[19] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. 2018. Microscopic traffic simulation using sumo. In *2018 21st international conference on intelligent transportation systems (ITSC)*. IEEE, 2575–2582.

[20] Branka Mirchevska, Christian Pek, Moritz Werling, Matthias Althoff, and Joschka Boedecker. 2018. High-level decision making for safe and reasonable autonomous lane changing using reinforcement learning. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2156–2162.

[21] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

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

[23] Sanmit Narvekar and Peter Stone. 2018. Learning curriculum policies for reinforcement learning. *arXiv preprint arXiv:1812.00285* (2018).

[24] Daniel E Rivera, Manfred Morari, and Sigurd Skogestad. 1986. Internal model control: PID controller design. *Industrial & engineering chemistry process design and development* 25, 1 (1986), 252–265.

[25] Ahmad El Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. 2016. End-to-end deep reinforcement learning for lane keeping assist. *arXiv preprint arXiv:1612.04340* (2016).

[26] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[27] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction.* MIT press.

[28] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. 2008. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of field Robotics* 25, 8 (2008), 425–466.

[29] Ardalan Vahidi and Azim Eskandarian. 2003. Research advances in intelligent collision avoidance and adaptive cruise control. *IEEE transactions on intelligent transportation systems* 4, 3 (2003), 143–153.

[30] Junjie Wang, Qichao Zhang, Dongbin Zhao, and Yaran Chen. 2019. Lane change decision-making through deep reinforcement learning with rule-based constraints. In *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–6.

[31] Pin Wang, Ching-Yao Chan, and Arnaud de La Fortelle. 2018. A reinforcement learning based approach for automated lane change maneuvers. In *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 1379–1384.

[32] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. 2016. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*. PMLR, 1995–2003.

[33] Ziran Wang, Guoyuan Wu, and Matthew J Barth. 2018. A review on cooperative adaptive cruise control (CACC) systems: Architectures, controls, and applications. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2884–2891.

[34] Jingda Wu, Zhiyu Huang, Wenhui Huang, and Chen Lv. 2022. Prioritized experience-based reinforcement learning with human guidance for autonomous driving. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[35] Yuyang Xia, Shuncheng Liu, Xu Chen, Zhi Xu, Kai Zheng, and Han Su. 2022. RISE: A Velocity Control Framework with Minimal Impacts based on Reinforcement Learning. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2210–2219.

[36] Jiechao Xiong, Qing Wang, Zhuoran Yang, Peng Sun, Lei Han, Yang Zheng, Haobo Fu, Tong Zhang, Ji Liu, and Han Liu. 2018. Parametrized deep q-networks learning: Reinforcement learning with discrete-continuous hybrid action space. *arXiv preprint arXiv:1810.06394* (2018).

[37] Zhi Xu, Shuncheng Liu, Ziniu Wu, Xu Chen, Kai Zeng, Kai Zheng, and Han Su. 2021. 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*. 2271–2280.

[38] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. 2020. A survey of autonomous driving: Common practices and emerging technologies. *IEEE access* 8 (2020), 58443–58469.

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