Transportation Research Part C 124 (2021) 102980

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Hybrid deep reinforcement learning based eco-driving for low-level connected and automated vehicles along signalized corridors

Qiangqiang Guo, Ohay Angah, Zhijun Liu, Xuegang (Jeff) Ban *

University of Washington, WA, United States

ARTICLE INFO

Keywords:
Connected and automated vehicles
Hybrid reinforcement learning
Policy gradient
Deep Q-learning
Eco-driving
Lane-changing

ABSTRACT

Eco-Driving has great potential in reducing the fuel consumption of road vehicles, especially under the connected and automated vehicles (CAVs) environment. Traditional model-based Eco-Driving methods usually require sophisticated models and cannot deal with complex driving scenarios. This paper proposes a hybrid reinforcement learning (RL) based Eco-Driving algorithm considering both the longitudinal acceleration/deceleration and the lateral lane-changing operations. A deep deterministic policy gradient (DDPG) algorithm is designed to learn the continuous longitudinal acceleration/deceleration to reduce the fuel consumption as well as to maintain acceptable travel time. Collecting the critic's value of each single lane from DDPG and integrating the information of adjacent lanes, a deep Q-learning algorithm is developed to make the discrete lane-changing decision. Together, a hybrid deep Q-learning and policy gradient (HDQPG) method is developed for vehicles driving along multi-lane urban signalized corridors. The method can enable the controlled vehicle to learn well-established longitudinal fuel-saving strategies, and to perform appropriate lane-changing operations at proper times to avoid congested lanes. Numerical experiments show that HDQPG can reduce fuel consumption by up to 46% with marginal or no increase of travel times.

1. Introduction

Improving the fuel efficiency of road vehicles has become a critical issue of the modern society since the transportation sector consumes about two thirds of the petroleum-based energy which has caused severe energy shortage and environmental problems (Borken et al., 2007; Yan et al., 2011). From the technology point of view, there are generally two types of approaches to reduce the fuel consumption of vehicles on roadways: (1) Develop more fuel-efficient vehicle technologies such as lightweight, hybridization, and advanced fuel-saving engines; and (2) Apply Eco-Driving strategies (and other traffic/transportation management strategies) so that vehicles can drive at economic speeds and make use of the kinetic energy to save fuels (Walnum and Simonsen, 2015). The former has been the focus of fuel-saving technologies for many years and now requires major breakthroughs in new material/mechanical/energy sciences for further improvements, which can be time and capital consuming. Recently, the latter, i.e., Eco-Driving, has received increasing attention due to its high fuel-saving potentials by only relying on existing vehicular technologies.

Narrowly speaking, Eco-Driving refers to reducing the energy consumption by improving the operations on steering wheel, acceleration pedal and brake pedal. Traditionally, Eco-Driving was implemented by driver education programs (Row, 2010; Barić et al., 2013), where drivers were taught about how to apply fuel-saving operations in their daily driving tasks. It was found that fuel consumption can be reduced right after the education program, which however may return back to normal in the long term

* Corresponding author.
E-mail address: banx@uw.edu (X. Ban).

https://doi.org/10.1016/j.trc.2021.102980
Received 10 August 2020; Received in revised form 7 January 2021; Accepted 10 January 2021
Available online 21 January 2021

due to human complacency and behavioral regression (Zarkadoula et al., 2007). In the last decades, the connected and automated vehicles (CAVs) have emerged and been rapidly developed, making it possible to operate Eco-Driving strategies automatically. CAVs can communicate with other vehicles (V2V), road side infrastructure (V2I), other road users such as pedestrians and bicyclists (V2X), and remote servers which can provide the information of surrounding vehicles, high-resolution maps, traffic signals, etc. Such information can be processed by vehicle control algorithms to generate Eco-Driving strategies, which can then be operated by the actuators (i.e., the automation feature of CAVs). The U.S. Department of Transportation's National Highway Traffic Safety Administration (NHTSA) defined five levels of automated driving, from driving assistance (Level 1) to fully automated vehicles (Level 5) (NHTSA, 2016). While low-level vehicle automation (e.g., Level 2) is maturing and increasingly deployed in vehicles, it is generally agreed now that it may take a relatively long time (10–20 years or more) for high-level vehicle automation (e.g., Level 4 or Level 5) to become available and widely deployed in vehicle fleets. The Eco-Driving methods presented in this paper only requires that the lateral lane-changing and longitudinal acceleration/deceleration operations can be controlled, which can be applied to low-level automated vehicles. This makes the proposed method practical in the near future.

CAV-based Eco-Driving techniques have usually been studied by model-based methods. In the literature, the Eco-Driving problem was often formulated as an optimal control problem (OCP) where fuel consumption and travel time were considered as the objective, and vehicle dynamics/signal plans/safety considerations were modeled as the constraints. Various solution methods such as dynamic programming (DP) and pseudo-spectral methods have been proposed to solve those OCPs. For example, for vehicles equipped with continuous variable transmission (CVT), Li and Peng (2012) solved the fuel-minimizing problem under the car-following scenario by Gauss pseudo-spectral method and found that the "pulse and glide" (PnG) operations performed the best. Later, researchers from the same group proposed the servo-loop PnG controllers for CVT vehicles (Li et al., 2012) and step-gear vehicles (Li et al., 2016), which showed great improvements for fuel savings. Hellström et al. (2009) developed a DP-based look-ahead control method for heavy trucks, which could make use of the road slope database and GPS information to generate fuel-efficient speeds. For urban scenarios that consist of signalized intersections, the OCPs become more complex due to additional constraints introduced by traffic signals. Rakha and Kamalanathsharma (2011) proposed a rule-based Eco-Driving model for a single vehicle driving across signalized intersections, assuming there were no surrounding vehicles. The proposed rules specified that: (1) If the current signal phase is green and the ego vehicle (i.e. the controlled vehicle) can accelerate to pass the intersection before the end of the green phase, it will accelerate to certain speed calculated by the kinetic equation; (2) If the ego vehicle has to decelerate to wait during the red phase, the speed profile of that deceleration process is optimized based on a VT-Micro emissions model. Xia et al. (2013) also developed a logic-based dynamic Eco-Driving speed planning method for vehicles driving along corridors with multiple lanes. The method dynamically generated acceleration and deceleration profiles with trigonometric increments to ensure smooth trajectories. Based on simulation studies on multi-lane corridors with different penetration rates and traffic volumes, the method was shown to be effective for fuel savings.

Readers can refer to Huang et al. (2018) for more detailed reviews on model-based Eco-Driving methods. In summary, these methods need to simplify the dynamics of the environment and usually can only deal with one specific driving scenario (e.g., car-following, intersection passing, etc.). The model can become more complex if the driving scenario are more complex, making it hard to be numerically solved. Current model-based methods usually adopt rule-based schemes to decompose the overall problem into sub-problems that can be solved individually. As a result, the accuracy and generalization ability of these methods also need further improvements.

Recently, learning-based methods, especially reinforcement learning (RL) based Eco-Driving methods, have gained traction due to their capability to overcome the above disadvantages. RL is an agent-oriented machine learning scheme based on the Markov decision process (MDP), where an agent learns to generate the optimal actions to maximize the cumulative rewards based on repeated interactions with the environment (Sutton and Barto, 2018). The action sequence under a series of states is the *policy* of the agent (see Section 2.1 for the precise definition of policy). In terms of improving the fuel efficiency, RL has been applied mainly in two fields: energy management system (EMS) and general Eco-Driving (GED), which are summarized in Table 1. Note that here the value-based RL methods refer to those that learn the value functions of either states or actions based on temporal difference learning, while policy-based RL methods directly learn or approximate the optimal policy.

EMS is mainly designed for hybrid electrical vehicles (HEVs) or plugin hybrid electrical vehicles (PHEVs), which can be considered as a special case of Eco-Driving regarding optimizing the energy flow inside a vehicle. For examples, Qi et al. (2016) proposed a tabular Q-learning based EMS for PHEVs to optimize the power-split control in real time. The speed of the ego vehicle, road grade, percentage of remaining time to the destination, battery pack's state-of-charge (SOC), and available charging gain of the upcoming charging station were used as the state. The discretized power supply level of the internal combustion engine was defined as the action. The reward function was designed to minimize the fuel cost while satisfying the power demand requirement. Numerical experiments with data synthesized from real-world traffic measurements showed that the proposed EMS algorithm could improve the fuel performance up to 12%. Researchers from the same group further modified this framework by adding a deep Q network (DQN) to better approximate the Q value (Qi et al., 2017, 2019). Hu et al. (2018) developed a deep Q-learning based EMS for HEVs. They used the total required torque and the SOC of battery as the state and discretized the output torque from the internal combustion engine as the action. Numerical experiments showed that the proposed EMS could outperform the rule-based EMS in terms of fuel savings.

EMS focuses on optimizing the internal energy flow of HEVs and PHEVs. The state is usually defined from the dynamics of the ego vehicle. GED, on the other hand, assumes there is an explicit relationship between the dynamics of the ego vehicle (e.g., longitudinal acceleration/deceleration) and energy consumption, and tries to optimize the dynamics of the ego vehicle given the information of the environment (e.g., surrounding vehicles, signals, etc.). In terms of GED, the ego vehicle is usually regarded as the agent; the

Table 1
Summary of current RL-based energy efficient studies.

	Papers	Vehicle type		Driving Scenario			RL algorithm			Action				
		ICV	HV	CF	SU	ST	VB	PB	HB	LO	LA	IN	CO	DI
EMS	Qi et al. (2016)	~	√	~	~	~	√	~	~	~	~	√	~	√
	Qi et al. (2017)	~	√	~	~	~	√	~	~	~	~	√	~	√
	Qi et al. (2019)	~	√	~	~	~	√	~	~	~	~	√	~	√
	Hu et al. (2018)	~	√	~	~	~	√	~	~	~	~	√	~	√
GED	Shi et al. (2018)	√	~	~	√	~	√	~	~	√	~	~	~	√
	Gamage and Lee (2016)	√	~	~	√	~	√	~	~	√	~	~	~	√
	Gamage and Lee (2017)	√	~	√	√	~	√	~	~	√	~	~	~	√
	Qu et al. (2020)	~	√	√	√	~	~	√	~	√	~	~	√	~
	Hao et al. (2020)	~	√	~	√	√	√	~	~	√	√	~	~	√
	Proposed method	√	~	~	√	√	~	~	√	√	√	~	√	√

ICV — internal combustion vehicles; HV — hybrid vehicles; CF — car-following; SU — signalized urban; ST — surrounding traffic; VB — value-based; PB — policy-based; HB — hybrid; LO — longitudinal; LA — lateral; IN — internal actions; CO — continuous; DI — discrete

longitudinal (i.e., acceleration and deceleration) and lateral (i.e., lane changing) operations of the ego vehicle can be modeled as actions; driving scenarios, including dynamics of surrounding vehicles and physical structure of intersections, can be considered as the environment; fuel consumption and travel time can be used as the rewards.

A few studies in recent years have tried to apply RL-based techniques to GED. For examples, Shi et al. (2018) proposed a Q-learning based Eco-Driving algorithm for a single vehicle driving across signalized intersections by assuming no surrounding vehicles. They predefined the actions (i.e., the longitudinal acceleration/deceleration) as discrete variables, and used the distance between the ego vehicle and the upcoming intersection, signal status, and instant vehicle speed as the state. The total carbon dioxide was used as the reward. The Eco-Driving strategies learned by Q-learning were found to be able to reduce the fuel consumption as well as to improve the traffic performance. The same framework was also used in Gamage and Lee (2016). Gamage and Lee (2017) extended this framework to the car-following scenario. The state was represented by the same three parameters in Shi et al. (2018) with an additional parameter, i.e., the headway between the ego vehicle and the front vehicle. The reward function was constructed as inversely proportional to the cumulative fuel consumption experienced by the ego vehicle between two successive decision points. However, the positive results found in the paper were built on the assumption that the front vehicle was under the free flow condition. Qu et al. (2020) developed a deep deterministic policy gradient based algorithm for electric CAVs under the car-following scenario to dampen stop-and-go waves and improve electric energy consumption. Experiment results showed that the proposed method could improve travel efficiency as well as reduce the average electric energy consumption. These algorithms only considered the longitudinal operations while neglecting the lateral lane-changing operation. Hao et al. (2020) proposed a deep Q-learning based Eco-Driving framework to deal with the multi-lane scenario where the ego vehicle was surrounded by human-driven vehicles. More specifically, they used camera data and signal information gathered from intelligent transportation systems (ITS) as states, and used the discretized longitudinal acceleration/deceleration, together with three discrete lane-changing indicators, as actions. They applied the dueling deep Q-network (DDQN) along with a long-short term reward (LSTR) function instead of the instant-variable to represent the reward function. Comparing their method with the intelligent driver model (IDM) and the Speed-First method, they showed that 12.25%–47.1% of energy savings can be achieved through this approach. In order to apply the value-based RL method, they discretized the continuous action space (i.e., longitudinal acceleration/deceleration). Although such discretization could make the value-based RL applicable, several disadvantages were also introduced: (1) the longitudinal acceleration/deceleration may not be smooth, which will increase the fuel consumption due to extra transient fuel consumption caused by abrupt changes of accelerations; and (2) the dimension of the discretized action space could be large such that the value-based RL algorithms may be computationally demanding or unable to converge.

In summary, studies that have applied RL-based methods for GED under signalized intersections and corridors are relatively sparse, which also suffer from a few issues. First, they usually over-simplified the problem by, e.g., assuming that there is only one vehicle driving along a single lane (Shi et al., 2018) or considering only the car-following scenario while neglecting the lane-changing behavior (Gamage and Lee, 2017; Qu et al., 2020). Second, studies that did consider surrounding vehicles and lateral operations, however, discretized the longitudinal acceleration/deceleration such that the value-based (e.g., Q-learning) RL algorithms can be applied (Hao et al., 2020), which may increase fuel consumption and is hard to converge when the dimension of the action space becomes large. This paper aims to fill these gaps by developing a new RL-based Eco-Driving model and its solution algorithm for both the longitudinal (i.e., acceleration/deceleration) and lateral (i.e., lane changing) operations of a CAV driving along signalized corridors, considering the impacts of surrounding vehicles. Unlike existing studies, we model the longitudinal acceleration/deceleration as continuous variables and the lateral lane changing as discrete variables, which is solved by a Hybrid Deep Q-learning and Policy Gradient (HDQPG) method. Numerical experiments conducted in this paper show that the proposed model outperforms state-of-the-art models in the literature. The main contributions of this paper are:

- We propose a hybrid modeling framework for the Eco-Driving problem for CAVs with low-level automation considering both continuous longitudinal acceleration/deceleration and discrete lateral lane-changing operations.

- Based on the hybrid modeling framework, we develop a hybrid RL algorithm (i.e. the HDQPG method) that integrates the deep deterministic policy gradient method (for longitudinal control) and deep Q-learning (for lateral control) to solve the proposed Eco-Driving model.
- We prove by simulation that the proposed model and solution method could improve the overall performance of a controlled CAV in terms of travel time and fuel consumption, and show that the learned fuel-saving strategies are consistent with those calculated by model-based methods.

This paper is organized as follows. We first define the Eco-Driving problem and illustrate the modeling framework in Section 2. We present the detailed model and solution method in Section 3. The results of numerical experiments are discussed in Section 4. Finally, Section 5 concludes this paper with discussions of future research.

2. Methodology overview

2.1. Preliminaries of RL

We first present some basics of the RL method, which will be helpful when we introduce the Eco-Driving scenario in detail in the next section. RL is a MDP where an agent learns to optimize his/her actions to maximize the cumulative reward based on repeated interactions with the environment. The agent observes current state and reward from the environment, and selects the optimal action based on the principle of maximizing the long-term cumulative reward. The environment receives and conducts the action, and generates new state and reward, which will be feedback to the agent. Consider a general RL problem: at time t, an agent observes the state $s_t \in \mathcal{S}$ from the environment, performs an action $a_t \in \mathcal{A}$ based on certain learning mechanism, and receives reward $r_{t+1} = r(s_t, a_t)$ when the environment moves to next state s_{t+1}. The decision sequence of the agent can be represented by the probability density of taking action a_t given state s_t, denoted as policy $\pi(a_t|s_t)$. Assuming that the agent takes action a_t under current state s_t, the probability of the environment transferring to s_{t+1}, denoted as $p(s_{t+1}|s_t, a_t)$, can represent the dynamics of this MDP. The agent tries to maximize the expected cumulative discounted reward $r_t^\gamma \doteq r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k})$, where γ is the discount rate with $0 \le \gamma \le 1$. The discounted reward represents the present value of future rewards. Adding the discounting rate γ can properly convert future rewards to current value, which also guarantees that the total discounted reward r_t^γ is bounded. The value of a state s_t under a policy π is the expected discounted reward when starting at state s_t and following the policy π. Hereafter, we denote it as the *state-value function* for policy π: $V_\pi(s_t) \doteq \mathbb{E}_\pi\left[r_t^\gamma|s_t\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k})|s_t\right]$. Similarly, the value of taking action a_t at state s_t under policy π is the expected discounted reward when starting at state s_t, taking action a_t, and following the policy π. Hereafter, we denote it as the *action-value (or Q-value) function* for policy π: $Q_\pi(s_t, a_t) \doteq \mathbb{E}_\pi\left[r_t^\gamma|s_t, a_t\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k})|s_t, a_t\right]$. Obviously, $V_\pi(s_t) = \sum_a \pi(a|s_t) Q_\pi(s_t, a)$.

2.2. Eco-driving problem

As shown in Fig. 1, we focus on controlling a CAV to drive across a signalized corridor along with other vehicles, aiming to minimize its fuel consumption and travel time. We call this vehicle the "ego" vehicle. In order to describe the studied problem more clearly, we make the following assumptions:

- The ego vehicle can get access to its own position/speed/acceleration from the controller area network (CAN) or on-board sensors.
- The ego vehicle can either detect the relative distance/speed/acceleration between itself and the surrounding vehicles by on-board sensors or be informed of such information by V2I communications. To be conservative, the detection distance is set to 200 m. This is practical under current onboard sensor techniques such as the millimeter wave radar.
- The longitudinal and lateral movements of the ego vehicle can be controlled automatically.
- Signal timing plans of the upcoming intersection can be sent to the ego vehicle through V2I communications.
- We consider multi-lane urban roadways where vehicles can change lanes during driving. For simplicity, three lanes are considered in this paper and the proposed RL framework can be properly modified for other multi-lane scenarios.

Modeled under the RL framework, the ego vehicle is the *agent* and other vehicles, signals, roads, and the traffic rules together constitute the *environment*. There are two types of *actions* that the agent can take, i.e., the longitudinal and lateral accelerations. One can model the action space by three approaches. First, one can define the action space as a two dimensional (i.e., lateral and longitudinal) continuous vector. Second, noticing that people usually take a series of fixed operations after they have decided to change lanes, one can model the action space as a discrete-continuous hybrid space, i.e., one continuous longitudinal action (acceleration/deceleration) and one discrete lateral action (change to the left lane, change to the right lane, or stay at current lane). Third, based on the second approach, one can further discretize the longitudinal acceleration/deceleration so that all action candidates are discrete. The first approach is closest to real-world operations. However, when it comes to the RL-based control, although it is suitable to model the lateral acceleration/deceleration by continuous variables for scenarios without traffic lanes (such as racing competition), for regular driving scenarios with dedicated lanes and traffic rules, such approach may introduce unpractical and even dangerous operations for lateral operations. For example, the ego-vehicle might be controlled to drive right above or very close to the lane marker for a relative long time, especially during the training process. Although one can design specific reward functions to penalize such "inappropriate" behavior, this could make the reward function in this paper more complex.

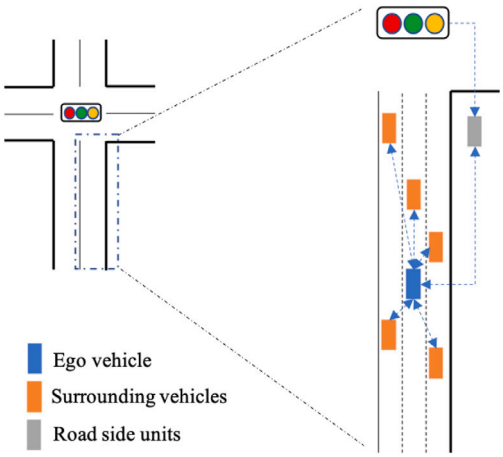

Fig. 1. Eco-Driving scenario.

In real-world driving, one usually takes a series of fixed actions to complete the lane-changing once s/he has decided to change lanes. Therefore, using discrete "decisions" to model the lateral operations may be a better choice. The third approach, as used in Hao et al. (2020), can simplify the action space and make it possible to apply one unified value-based RL algorithm to deal with the Eco-Driving problem. However, as discussed in Section 1, it suffers from extra fuel consumption and poor convergence of the algorithm. The second approach can be interpreted as a combination of the first and third approaches, which still treats the longitudinal acceleration/deceleration as continuous, while using discrete variables to represent whether the ego vehicle should change lane and if so which lane should be targeted. In this way, the undesired lane changing behavior as described above for the first approach can be avoided. On the other hand, the continuous longitudinal acceleration/deceleration makes it possible to generate smooth acceleration/deceleration trajectories to reduce fuel consumption. Therefore, we adopt the second approach to model the action space in this paper.

By using the discrete-continuous hybrid action space, the Eco-Driving problem can be decomposed into two sub-problems: lateral lane-changing (i.e., deciding to stay on the current lane or change to the right/left lane if needed) and single lane longitudinal operation (i.e., longitudinal acceleration/deceleration). We develop a hybrid framework to model and solve this problem. At a specific time, we first extract information of the lane where the ego vehicle resides (called the "ego lane"), which is used as the state of the longitudinal RL controller (see Section 3.1.1). The RL controller generates longitudinal actions (i.e., longitudinal acceleration/deceleration) given current state s, which will be applied to the environment. We then assume that there are also two "virtual" ego vehicles, one for each of the two neighboring lanes with the same vehicle state as the real ego vehicle (i.e., distance to next intersection, speed, and acceleration). Similar to the state of the ego lane, the states of two neighboring lanes with the virtual vehicles can also be extracted (see Section 3.1.1 and Section 3.2.1). Note that although the states of the virtual ego vehicles are the same as the real ego vehicle, the full state of each neighboring lane is different due to different states of the following vehicles (see Section 3.1.1 for the state definition). We then calculate the critic's value of each of the three lanes (see Section 3.1.3) and use them as part of the state representations for the lateral lane-changing RL controller. Additional information that are useful but excluded from the single lane state (e.g., that related to the following vehicles in two neighboring lanes) are also gathered (see 3.2.1). Integrating the critic's values of the three lanes and additional lane-changing information, we build the state for the lateral lane-changing RL controller (see 3.2.1). Given such state, the lateral RL controller decides whether the ego vehicle should change to the right lane, change to the left lane, or stay on the current lane. Compared with directly using information of surrounding vehicles of the ego vehicle as the state of the lateral RL controller, the proposed state representation, by introducing the concept of virtual ego vehicles, can help reduce the dimension of the state space; see Section 3.2.1 for more discussions.

It should be noted that the actions generated by the RL algorithms may not be achievable under the corresponding driving conditions. For example, the lateral RL decides to change to the left lane but it is too close to the front (or following) vehicle on the left lane (also reflected by the distance between the virtual ego vehicle and the front or following vehicle on that lane). To ensure safety, one can design safety-guaranteed RL algorithms to always generate safe decisions or behaviors. This however needs sophisticated RL design schemes such as dynamic output constraints, which can make the RL algorithm very complex. In this paper, we develop a "checking-feedback-learning" (CFL) framework to not only ensure safe and consistent driving behaviors but also encourage the RL algorithms to learn safe and consistent behaviors via discouraging the learning of unsafe or inconsistent behaviors. Given the actions generated by the RL algorithm, CFL conducts safety (or consistency) checks and will not conduct the actions if they are not safe or inconsistent with the original decision. Meanwhile, if an unsafe or inconsistent behavior is detected, a penalty term will be added to the reward (i.e., Eqs. (3) and (9)) to penalize the unsafe/inconsistent behavior. In this way, the CFL framework ensures safety and encourages the RL algorithms to learn safe driving behaviors and consistent decisions. In the following section, we show this hybrid RL framework in detail.

3. Hybrid deep Q-learning and policy gradient (HDQPG) model for eco-driving

3.1. The longitudinal RL model

In this section, we discuss the longitudinal RL model, focusing on its four key components: state, action, reward, and the learning algorithm.

3.1.1. State and action

At time t, the distance between the ego vehicle and the next traffic signal (d_t^e), ego vehicle's speed (v_t^e) and acceleration (acc_t^e), distance difference (Δd_t^e), speed difference (Δv_t^e), and acceleration difference (Δa_t^e) between the ego vehicle and the front vehicle, the time to next green phase of the traffic signal (ΔT_t, $\Delta T_t = 0$ if the current phase is green), and the time duration of the next green phase or the remaining green if current phase is green (T_t^g) are used as the state. These variables are selected based on the real driving experience and the assumed technologies (i.e., V2I and the sensing capability of CAVs): when one drives along a one-lane road (as assumed by the longitudinal model), the information s/he usually uses to make driving decisions is related to the ego vehicle, the front vehicle, and the traffic signal. In this paper, we assume that the ego vehicle knows which lane it drives on. This is achievable given current localization and multi-sensor fusion techniques. We assume that the geometry information such as the distances between adjacent traffic signals of the corridor is known. And the ego vehicle could obtain the traveled distance from the last traversed traffic signal to its current position, which can be considered as the relative longitudinal lane-level position. d_t^e is then calculated by subtracting the traveled distance from the distance between the two traffic signals. Other variables, such as the lateral lane-level positions (i.e. the distances between the ego-vehicle and the front/following vehicles of adjacent lanes), can be directly collected by on-board sensors and roadside units. The above identified variables constitute a 8-dimension vector that is used here as the state of the longitudinal RL model, i.e., $s_t^{\text{lg}} = (d_t^e, v_t^e, acc_t^e, \Delta d_t^e, \Delta v_t^e, \Delta a_t^e, \Delta T_t, T_t^g)$. Naturally, the longitudinal acceleration/deceleration serves as the action a_t^{lg} of the single lane Eco-Driving algorithm, which is continuous and bounded.

3.1.2. Reward

In order to achieve the Eco-Driving task, three aspects should be considered when designing the reward:

- First, the fuel consumption should be reduced. Therefore, we add the fuel consumption at the current time step g_t, multiplied by a negative weight $-w_f$, to the reward.
- Second, if minimizing fuel consumption is the only objective, the optimal action of the ego vehicle would be stopping at the origin with the engine shut down. Thus, in order to fulfill the travel needs, travel time should be kept at an acceptable level. The total travel time, which can only be calculated at the end of the travel, is a delayed reward and is hard to be distributed to each step of the whole MDP. In this paper, we use the travel distance (l_t) as a proxy to reflect the travel time. For each time step, since the elapsed time is fixed, the longer the travel distance is, the shorter the travel time would be.
- Third, the CFL framework discussed earlier is applied to ensure driving safety (e.g., following traffic lights and avoiding collisions) for longitudinal control. To do so, we introduce the concept of "safety speed" $v_{\text{safe}}^e(t)$, which is the maximum speed that the ego vehicle can drive at time t without breaking any safety constraint. We first calculate the expected speed $\tilde{v}_t^e$ that the ego vehicle could achieve given action a_t at current speed v_t^e (i.e., $\tilde{v}_t^e = v_t^e + a_t^{\text{lg}} \times \Delta t$). We check if the expected speed is larger than the safety speed (i.e., $\tilde{v}_t^e > v_{\text{safe}}^e(t)$), which means the safety constraints are violated. If so, we will modify the action (i.e, original a_t^{lg}) to $(v_{\text{safe}}^e(t) - v_t^e)/\Delta t$ to make sure the actual action is "safe". Meanwhile, in this case, we will add the difference between the expected speed and the safety speed (i.e., $\tilde{v}_t^e - v_{\text{safe}}^e(t)$) to the reward with a negative weight (i.e., $-w_s$) to penalize such unsafe actions; see Eq. (3) later. This penalty term provides a feedback to the RL algorithm to encourage it to learn safe driving behaviors.

Fig. 2 shows the logic of how to calculate the safety speed $\tilde{v}_t^e$. The basic idea is to evaluate the traffic conditions in front of the ego vehicle. Under the simplest condition, where there are no upcoming traffic signal in sight (or there is a signal but with green phase) and no leading vehicle, we simply set the safety speed as the speed limit of the current road. If there is a traffic signal in sight, we check the phase of the upcoming signal. If the phase is red, or is yellow but the ego vehicle can brake to stop with maximum deceleration, we will create a virtual stopped leading vehicle (i.e., a leading vehicle with 0 speed) at the signal position. Meanwhile, we check if there is a real leading vehicle. If yes, we take the closest one from the virtual vehicle and the real leading vehicle as the selected leading vehicle for the purpose of calculating the safety speed $\tilde{v}_t^e$. We then collect the current action a_t, current speed of the ego vehicle v_t^e, current speed of the selected leading vehicle $\hat{v}_t^l$, and the gap between the ego vehicle and selected leading vehicle Δd_t, and use these variables to calculate the safety speed by the Krauss car-following model (Krauß et al., 1997; Krauß, 1998). Essentially, the safety speed of Krauss model is calculated as

$$v_{\text{safe}}^{\text{Krauss}}(t) = \hat{v}_t^l + \left(\Delta d_t - \hat{v}_t^l \tau\right) / \left(\frac{\hat{v}_t^l + \hat{v}_t^e}{2a_{\max}} + \tau\right) \tag{1}$$

where τ is the human reaction time and $a_{\max}$ is the maximum deceleration of the ego vehicle. Note that τ is used here even we assume the ego vehicle is a CAV, in order to produce more conservative safety speeds. In addition, the speed should be bounded by the speed limit. So we take the minimum of the Krauss safety speed and the speed limit as the final safety speed under such conditions.

$$v_{\text{safe}}^e(t) = \min\{v_{\max}, v_{\text{safe}}^{\text{Krauss}}(t)\} \tag{2}$$

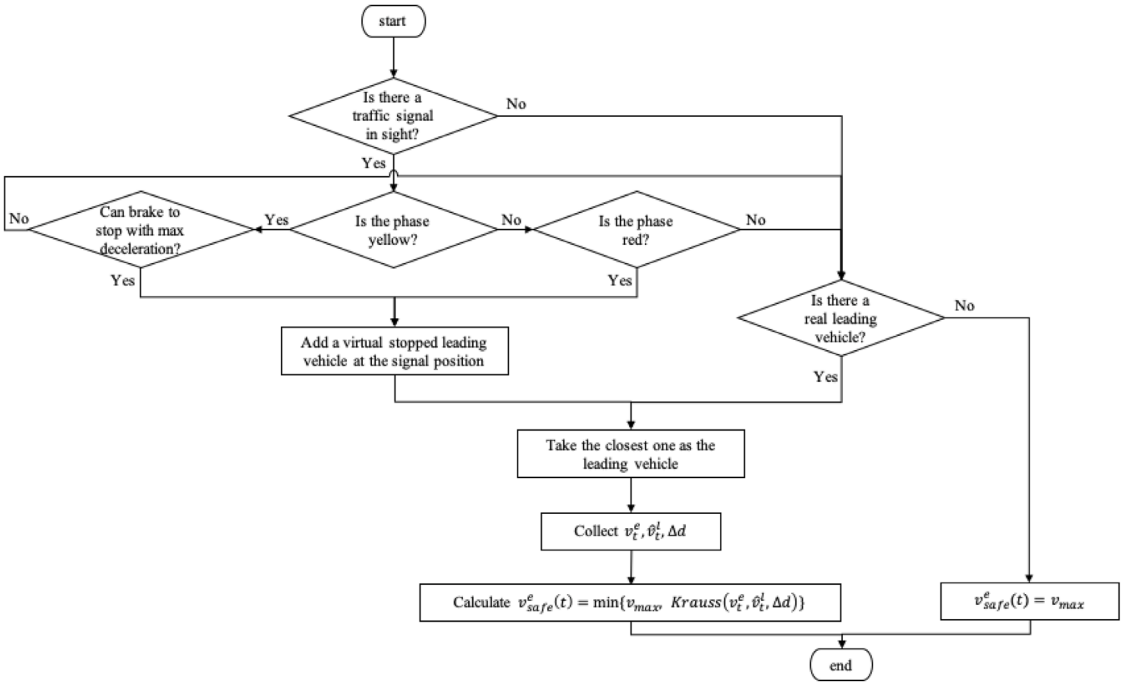

Fig. 2. Flowchart of calculating the safety speed.

In summary, the reward function for time step t, r_t^{lg}, can be expressed as Eq. (3). Note that since r_t^{lg} represents reward, the sign for each term needs to be set properly. Also in order to make these three different terms comparable as well as to adjust our preference, we add two weight parameters w_f and w_s to fuel consumption and safety, respectively. These two weights can be interpreted as two parameters that convert fuel consumption and safety to "distance".

$$r_t^{\text{lg}} = -w_f g_t + l_t - w_s \max\{0, \tilde{v}_t^e - v_{\text{safe}}^e(t)\} \tag{3}$$

It should be noted that there are other alternatives to balance the fuel consumption and travel time in the reward function, e.g., considering the distance normalized fuel consumption or a target-oriented reward. Detailed discussions are omitted here to save space. Based on Eq. (3), the discounted cumulative reward that is used in the longitudinal RL algorithm can then be defined as

$$r_t^{\text{lg},\gamma} = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}^{\text{lg}} \tag{4}$$

3.1.3. Deep deterministic policy gradient (DDPG)

The longitudinal Eco-Driving model can be regarded as an MDP with a continuous action space (i.e., the longitudinal acceleration/deceleration). We apply the policy gradient (PG) based RL method (Sutton and Barto, 2018) to solve such a model. From the policy point of view, the ego vehicle's goal is to find a policy $\pi(a_t^{\text{lg}}|s_t^{\text{lg}})$ that can maximize the cumulative discounted reward from the start state s_0^{lg}: $J(\pi) = \mathbb{E}[r_0^{\text{lg},\gamma}|\pi]$. Similar to gradient methods in traditional optimization theory, policy gradient aims to find the optimal policy by improving the policy using gradients generated from the change of objectives. Note that for the sake of clarity, in this subsection (i.e., 3.1.3), we will not include the indicator "lg" in the notations, i.e., $s_t^{\text{lg}}, a_t^{\text{lg}}, r_t^{\text{lg}}$ and $r_t^{\text{lg},\gamma}$ will be represented by s_t, a_t, r_t and r_t^{γ} respectively. In this way, the policy $\pi(a_t^{\text{lg}}|s_t^{\text{lg}})$ and the cumulative reward $J(\pi) = \mathbb{E}[r_0^{\text{lg},\gamma}|\pi]$ are represented by $\pi(a_t|s_t)$ and $J(\pi) = \mathbb{E}[r_0^{\gamma}|\pi]$ respectively. For continuous or large state/action spaces, it is inefficient or even impossible to calculate every $\pi(a_t|s_t)$ tabularly. Thus, $\pi(a_t|s_t)$ is usually approximated by a parameterized function $\pi_\theta(a_t|s_t) \sim \pi(a_t|s_t)$. In this section, we use θ as a common parameter notation for the approximation functions of both the stochastic policy π_θ and the deterministic policy μ_θ (which will be introduced later). The performance function then becomes $J(\pi) \sim J(\pi_\theta)$. Let $p(s \to s', k, \pi_\theta)$ be the probability of going from state s to state s' after k steps under policy π_θ. Define the discounted state distribution $\rho_{\pi_\theta}(s') = \int_S p_1(s) \sum_{k=0}^{\infty} \gamma^k p(s \to s', k, \pi_\theta)$ where p_1 is the initial state distribution. The performance function can be expressed as

$$J(\pi_\theta) = \int_S \rho^{\pi_\theta}(s) \int_{\mathcal{A}} \pi_\theta(a|s) r(s,a) \mathrm{d}a \mathrm{d}s \tag{5}$$

According to the policy gradient theorem (Sutton and Barto, 2018), we can calculate the partial derivative of the performance function

$$\begin{aligned}\nabla_\theta J\left(\pi_\theta\right) &= \int_S \rho^{\pi_\theta}(s)\int_A \nabla_\theta \pi_\theta(a|s) Q^{\pi_\theta}(s,a)\mathrm{d}a\mathrm{d}s \\ &= \mathbb{E}_{s\sim\rho^{\pi_\theta},a\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a|s) Q^{\pi_\theta}(s,a)\right]\end{aligned} \tag{6}$$

Eq. (6) is called the stochastic policy gradient since π_θ is a probability density function. Practically, the policy usually becomes deterministic as the policy gradient algorithm converges. However, the policy gradient $\nabla\pi_\theta$ fluctuates more rapidly near the mean, making it hard to estimate the stochastic policy (Zhao et al., 2011). To overcome this disadvantage, Silver et al. (2014) proposed the deterministic policy gradient algorithm (DPG) by introducing a deterministic policy $\mu_\theta(s): S \to \mathcal{A}$ parameterized with $\theta \in \mathbb{R}^n$. The performance function becomes

$$J\left(\mu_\theta\right) = \int_S \rho^{\mu_\theta}(s) r(s,\mu_\theta(s))\mathrm{d}s \tag{7}$$

And the deterministic gradient theorem shows that if $p\left(s'|s,a\right)$, $\nabla_a p\left(s'|s,a\right)$, $\mu_\theta(s)$, $\nabla_\theta\mu_\theta(s)$, $r(s,a)$, $\nabla_a r(s,a)$, $p_1(s)$ are continuous for all parameters, we have

$$\begin{aligned}\nabla_\theta J\left(\mu_\theta\right) &= \int_S \rho^{\mu_\theta}(s)\nabla_\theta\mu_\theta(s)\nabla_a Q^{\mu_\theta}(s,a)\Big|_{a=\mu_\theta(s)}\mathrm{d}s \\ &= \mathbb{E}_{s\sim\rho^{\mu_\theta}}\left[\nabla_\theta\mu_\theta(s)\nabla_a Q^{\mu_\theta}(s,a)\big|_{a=\mu_\theta(s)}\right]\end{aligned} \tag{8}$$

Given the above theories, the policy gradient can be implemented by various methods, of which the actor–critic is a widely used scheme. The actor tries to find the optimal policy using the policy gradient (i.e., Eq. (6) or (8)). The critic estimates the action value $Q^{\pi_\theta}(s,a)$ or $Q^{\mu_\theta}(s,a)$ by certain policy evaluation algorithms such as the Monte Carlo method and the temporal differential method. Note that $Q^{\pi_\theta}(s,a)$ and $Q^{\mu_\theta}(s,a)$ are the true action value functions, and we use a parameterized function $Q^{\omega}(s,a)$ to approximate the action value function. Examples of $Q^{\omega}(s,a)$ can be a linear model or a neural network where ω represents their parameters. Silver et al. (2014) proved that if $Q^{\omega}(s,a)$ is of the form $Q^{\omega}(s,a) = \left(a-\mu_\theta(s)\right)^{\top}\nabla_\theta\mu_\theta(s)^{\top}w + V^{v}(s)$ (where $V^{v}(s)$ is any differentiable baseline function that is independent of the action a) and ω minimizes the mean square error between $Q^{\omega}(s,a)$ and $Q^{\mu_\theta}(s,a)$, the function approximator $Q^{\omega}(s,a)$ is compatible, i.e., the gradient $Q^{\mu_\theta}(s,a)$ can be replaced by $Q^{\omega}(s,a)$ without affecting the deterministic policy gradient. In real world problems, however, non-linear approximators are usually needed especially in order to learn and generalize under large state spaces. To solve this problem, Lillicrap et al. (2015) proposed the deep deterministic policy gradient (DDPG) by using neural networks to approximate the action value function for the critic. Previous research results showed that the DDPG could find policies as good as those by algorithms with full access to the dynamics and its derivatives, which means that DDPG could learn policies that are close to the global optimal policies in an "end-to-end" manner. In order to capture the potential nonlinearity of the actor's policy function and the critic's value function, we adopt the DDPG method in this paper. In particular, we design two deep neural networks for the actor and critic separately. In addition, similar to Lillicrap et al. (2015), two soft target networks are used to stabilize the learning process. Given current single lane state s_t, DDPG directly generates the action $a_t = \mu_\theta(s_t)$ (i.e., longitudinal acceleration/deceleration) for next time step. Meanwhile, DDPG calculates the critic's value $Q^{\omega}(s_t,\mu_\theta(s_t))$, which represents the expected cumulative future reward under current state s_t and action a_t. Notice that this critic's value will also be used in the lateral RL model to reduce the dimension of the state space, which will be presented in detail in the next section.

3.2. The lateral RL model

3.2.1. State, action and reward

The most intuitive way to construct the state for the lateral lane-changing RL model is to directly extract the information of surrounding vehicles, i.e., the position, speed, and acceleration of the vehicles around the ego vehicle. Usually, information of at least five vehicles should be collected: the front vehicle on the ego lane, and the front and following vehicles of the other two neighboring lanes. Adding more information of the ego vehicle and traffic signal (i.e., time to the next green time and the duration of the green time), the dimension of such a state space is 5×3 (surrounding vehicles) $+3$ (ego vehicle) $+2$ (traffic signal) $= 20$. In this paper, we reduce the dimension of the state space by introducing a virtual ego vehicle to each of the two neighboring lanes (see also Section 2.2). For each lane, we can construct the single-lane state as discussed in 3.1.1. Applying the longitudinal DDPG algorithm to each of the three lanes, we can obtain: (i) the longitudinal action for the next time step based on the state of real ego lane; and (ii) the critic's value for all three lanes. As discussed in Section 3.1.3, the critic's value represents the potential cumulative reward of each lane, determined by the (virtual) ego vehicle and the front vehicle. We thus use the critic's value of each lane to replace the states of the front vehicle and the (virtual) ego vehicle in the state representation of the lateral RL controller. This results in four variables to represent the state of each of the two neighboring lanes, i.e., the critic's value and the position/speed/acceleration difference between the virtual ego vehicle and the following vehicle. For the ego lane, only the critic's value is used. Therefore, by introducing the virtual ego vehicles, the dimension of the state space for the lateral RL model reduces to 4×2 (non-ego lanes) $+1$ (ego lane) $= 9$, about half of the dimension of the state space if virtual ego vehicles were not used. It should be noted that adding more information from more surrounding vehicles might enable the ego vehicle to learn more useful maneuvers. However, collecting data beyond the above-discussed five vehicles is usually much harder than just collecting data from these five vehicles, especially under current

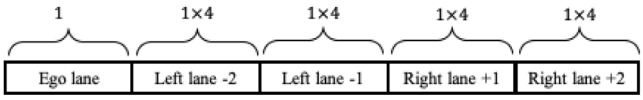

Fig. 3. State representation of lateral RL controller.

vehicle and traffic technologies. Therefore, we choose to use only the data of the five neighboring vehicles for lateral control in this paper.

The lateral RL controller should be able to identify from the state vector which part is for the information of the ego vehicle, which parts are for the information of surrounding vehicles, and what is the relative location among them. The first two are easy to implement. For example, one can use a vector with nine (9) elements as the state, where the first element represents the ego lane's information, and the rest eight (8) elements represent the information of the other vehicles. However, this approach cannot indicate in which lane the ego vehicle resides. To do so and to capture all possible ego lane location scenarios (i.e., left most, middle, right most), we design an encoding method shown in Fig. 3 for the state space representation. As aforementioned, the state of the ego lane consists of a single variable (i.e., the critic's value of the ego lane), which is used as the first segment. Then, we add four segment holders for the rest of the two lanes (the dimension of each of the segments is four). The number $(-2, -1, 1, 2)$ indicates the relative position of the segment (lane) with respect to the ego-lane: positive means to the right and negative means to the left. If the ego vehicle locates at the left most lane, which means both the other two lanes are to its right, we place the state of the adjacent right lane to the segment "Right lane +1" and place the state of the right most lane to the segment "Right lane +2". If the ego vehicle locates at the middle lane, we place the state of the adjacent right lane to the segment "Right lane +1" and place the state of the adjacent left lane to the segment "Left lane −1". If the ego vehicle locates at the right most lane, we place the state of the adjacent left lane to the segment "Left lane −1" and place the state of the left most lane to the segment "Left lane −2". In this way, the relative position of the ego lane and the other two lanes can be compatibly encoded. Notice that although there are four segments (in addition to the first one for the ego-vehicle), for a given scenario, only two of them will be occupied which represent the states for the two neighboring lanes. That is, the state space has a dimension of nine. It is also clear that if there are more or fewer than three lanes, the encoding method shown in Fig. 3 can still apply with straightforward modifications.

If the lateral RL controller generates a lane-changing decision to "change to the left lane" or "change to the right lane", we will first check whether such a decision can be completed or not, i.e., whether the traffic conditions allow the ego vehicle to change to the target lane safely. If the lane-changing decision is allowed, we will conduct it in the following 3 s (we assume that lane-changing can be completed in 3 s with a series of fixed operations). If the lane-changing decision is not allowed since it violates traffic rules or not safe (e.g., the distance to the front or following vehicle on the target lane is too short), the ego vehicle will be kept on the current lane. Meanwhile, a penalty term is added to the reward in this case (see the last term in (9) below), which provides feedback to the RL algorithm to discourage its learning of this unsafe behavior (or encourage the RL algorithm to learn safe behaviors). If the lane-changing decision is allowed, the 3 s after the lane-changing decision will be a "protected" period. If, subsequently, the RL controller generates any decision that conflicts with the original lane-changing decision during this protected period, the new decision will be regarded as not allowed. If this occurs, we will keep completing the original lane-changing decision and ignore the new decision. Meanwhile, we also add a penalty term to the reward to provide feedback to the RL algorithm which discourages its learning of the inconsistent decisions. As discussed in Section 2.2, here we apply the CFL framework to ensure safety and encourage the lateral RL algorithm to learn safe behavior and consistent decisions.

Let A be the set of events that the lane-changing decisions cannot be completed, i.e., A represents lane-changing scenarios where the lane-changing decisions generated by the lateral RL controller are not allowed considering real traffic conditions or not consistent with the original decision during the 3-second protected period. We can define the following reward function for time step t:

$$r_t^{\text{la}} = -w_f g_t + l_t - w_l \times \mathbf{1}_A(x_t) \tag{9}$$

where $\mathbf{1}_A$ is an indicator function (i.e., $\mathbf{1}_A(x) = 1$ if $x \in A$ and $\mathbf{1}_A(x) = 0$ if $x \notin A$). x_t is current lane-changing scenario, w_l is a penalty for generating lane-changing decisions that cannot be achieved, which discourages the RL algorithm to learn unsafe or inconsistent lane-changing decisions/behaviors. Similar to the longitudinal RL controller, the discounted cumulative reward for lateral algorithm is then defined by

$$r_t^{\text{la},\gamma} = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}^{\text{la}} \tag{10}$$

3.2.2. Deep Q-learning

The lateral lane-changing problem has a discrete action space consisting of three variables. We use Q-learning (Watkins and Dayan, 1992) as the basis to solve the lateral RL model due to its success in dealing with RL problems with small dimension discrete action spaces. At current state s_t (which is defined in Section 3.2.1), the agent estimates values of taking every possible action a_t (i.e, change to the left lane, change to the right lane, or stay at current lane), and selects the action that can maximize the long-term reward defined above in Eq. (10). The Q-values are learned iteratively by

$$Q_\pi(s_t, a_t) \leftarrow Q_\pi(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_a Q_\pi(s_{t+1}, a) - Q_\pi(s_t, a_t)) \tag{11}$$

where α denotes the learning rate. Notice that s_t, a_t, r_{t+1} in Eq. (11) are actually $s_t^{\text{la}}, a_t^{\text{la}}, r_{t+1}^{\text{la}}$ respectively. For the sake of clarity, we do not include the indicator "la" in the notation in this subsection. Tabular Q-learning can give satisfactory control orders if the state and action spaces are small. However, when dealing with large or continuous state and action spaces, Q-learning becomes quite inefficient or even unrealistic due to the large amount of memory required to store/update the table and the large amount of time required to explore possible states. Thus, function approximation methods such as those using deep neural networks (i.e., the deep Q-network, DQN) to estimate Q-values have been developed (Mnih et al., 2015) to solve this problem. In DQN, the Q-values are approximated by a parameterized neural network (denoted as $Q_\pi(s_t, a_t|\theta^Q)$). The update of Q-values (i.e., Eq. (11)) then becomes the update of network parameters θ^Q by minimizing the loss function:

$$L(\theta^Q) \approx (r_{t+1} + \gamma \max_a Q_\pi(s_{t+1}, a|\theta^Q) - Q_\pi(s_t, a_t|\theta^Q))^2 \tag{12}$$

There are several key considerations in DQN. First, when optimizing the network parameters θ, most optimization algorithms require that the samples should be independently distributed. The transitions collected along an MDP cannot directly satisfy this requirement due to the temporal correlation. In DQN-based RL, it is common to use a replay buffer to solve this problem. The replay buffer is a memory buffer with finite fixed spaces. During the learning process, the transitions $(s_t, a_t, r_{t+1}, s_{t+1})$ are collected and stored in the replay buffer. When the replay buffer is full, the oldest ones will be replaced by the latest ones. At each training step, a mini-batch is sampled uniformly from the replay buffer to update the DQN. The advantage of using replay buffer is that one can make efficient use of the stored samples and accelerate hardware optimizations (Lin, 1992). Second, the learning process is proved to be unstable if directly implementing Eq. (12), because $Q_\pi(s_t, a_t|\theta^Q)$ is used for both calculating the target Q-values for the next step and updating the current Q-network, which may result in fluctuations and even divergence (Tsitsiklis and Van Roy, 1996). In this paper, we create a target network $Q'_\pi(s_t, a_t|\theta^{Q'})$ to calculate the target Q-values (Mnih et al., 2013; Lillicrap et al., 2015). The loss function (i.e., Eq. (12)) then becomes

$$L(\theta^Q) \approx (r_{t+1} + \gamma \max_a Q'_\pi(s_{t+1}, a|\theta^{Q'}) - Q_\pi(s_t, a_t|\theta^Q))^2 \tag{13}$$

The parameter of the target network $\theta^{Q'}$ is soft-updated by $\theta^{Q'} \leftarrow \eta\theta^Q + (1-\eta)\theta^{Q'}$ where $\eta \leq 1$, which means the target network will slowly track the learned network regulated by η. This method has been shown to be able to stabilize the learning process (Lillicrap et al., 2015). For the Eco-Driving problem in this paper, considering that the state of the lateral lane-changing problem is continuous (i.e., the state space is discrete with 9 elements, while each state (e.g. the critic's value of the ego-vehicle) is continuous), we use two deep neural networks (i.e., one current network and one target network) to approximate the Q-values (see Section 4.2 for more details).

3.3. HDQPG

Combining the longitudinal DDPG and lateral DQN, we propose the HDQPG algorithm for Eco-Driving. The pseudo codes are shown in Algorithm . Note that in this algorithm we add "lg" and "la" back to indicate longitudinal and lateral variables respectively.

Algorithm 1 Hybrid deep Q-learning and policy gradient (HDQPG)

Randomly initialize the longitudinal Q network (critic) $Q_{\text{lg}}(s_{\text{lg}}, a_{\text{lg}}|\theta^{Q_{\text{lg}}})$ and π network $\pi_{\text{lg}}(s_{\text{lg}}|\theta^{\pi_{\text{lg}}})$, and lateral Q network $Q_{\text{la}}(s_{\text{la}}, a_{\text{la}}|\theta^{Q_{\text{la}}})$.
Initialize target network Q'_{lg}, π'_{lg}, and Q'_{la} with $\theta^{Q'_{\text{lg}}} = \theta^{Q_{\text{lg}}}, \theta^{\pi'_{\text{lg}}} = \theta^{\pi_{\text{lg}}}, \theta^{Q'_{\text{la}}} = \theta^{Q_{\text{la}}}$.
Initialize the replay buffer for longitudinal DDPG: $R_{\text{lg}} = \emptyset$ and the replay buffer for lateral DQN: $R_{\text{la}} = \emptyset$.
for episode = 1:Maximum number of episodes **do**
  Select $\varepsilon = \max(1 - \text{episode}/200, 0.01)$.
  Initialize two random processes $\mathcal{N}^{\text{lg}}$ for longitudinal action exploration.
  for $t = 1, T$ **do**
    Select lateral action $a_t^{\text{la}} = \text{argmax}_{a_t^{\text{la}}} Q_{\text{la}}(s_t^{\text{la}}, a_t^{\text{la}}|\theta^{Q_{\text{la}}})$ with probability (1-ε) or select lateral action randomly from $\{-1, 0, 1\}$ with probability ε, where $\{-1, 0, 1\}$ represent right turn, stay on current lane, and left turn, respectively.
    Select longitudinal action $a_t^{\text{lg}} = \pi_{\text{lg}}(s_t^{\text{lg}}|\theta^{\pi_{\text{lg}}}) + \mathcal{N}_t^{\text{lg}}$.
    Execute $\{a_t^{\text{la}}, a_t^{\text{lg}}\}$ and observe r_t^{lg}, r_t^{la}, and $s_{t+1}^{\text{lg}}, s_{t+1}^{\text{la}}$.
    Store transition $\left(s_t^{\text{lg}}, a_t^{\text{lg}}, r_t^{\text{lg}}, s_{t+1}^{\text{lg}}\right)$ to R_{lg}, remove the first transition if len(R_{lg}) > memory capacity of DDPG
    Store transition $\left(s_t^{\text{la}}, a_t^{\text{la}}, r_t^{\text{la}}, s_{t+1}^{\text{la}}\right)$ to R_{la}, remove the first transition if len(R_{la}) > memory capacity of DQN
    if len(R_{lg}) == memory capacity of DDPG **then**
      Sample a random mini-batch of m transitions from R_{lg}.
      Set $y_i = r_i^{\text{lg}} + \gamma Q'_{\text{lg}}(s_{i+1}^{\text{lg}}, \pi'_{\text{lg}}(s_{i+1}^{\text{lg}}|\theta^{\pi'_{\text{lg}}})|\theta^{Q'_{\text{lg}}})$.
      Update the longitudinal critic by minimizing the loss
      $L = \frac{1}{N}\sum_i \left(y_i - Q_{\text{lg}}(s_i^{\text{lg}}, a_i^{\text{lg}}|\theta^{Q_{\text{lg}}})\right)^2$.
      Update the actor policy by

$$\nabla_{\theta^{Q_{\mathrm{lg}}}} J \approx \frac{1}{N}\sum_i \nabla_{a_{\mathrm{lg}}} Q_{\mathrm{lg}}\left(s_{\mathrm{lg}}, a_{\mathrm{lg}}|\theta^{Q_{\mathrm{lg}}}\right)\Big|_{s_{\mathrm{lg}}=s_i^{\mathrm{lg}}, a_{\mathrm{lg}}=\pi_{\mathrm{lg}}\left(s_i^{\mathrm{lg}}\right)} \nabla_{\theta^{Q_{\mathrm{lg}}}} \pi_{\mathrm{lg}}\left(s_{\mathrm{lg}}|\theta^{Q_{\mathrm{lg}}}\right)\Big|_{s_i^{\mathrm{lg}}}$$

Update the target networks $\theta^{Q'_{\mathrm{lg}}} \leftarrow \tau\theta^{Q_{\mathrm{lg}}} + (1-\tau)\theta^{Q'_{\mathrm{lg}}}$, $\theta^{\pi'_{\mathrm{lg}}} \leftarrow \tau\theta^{\pi_{lg}} + (1-\tau)\theta^{\pi'_{\mathrm{la}}}$

end if

if len(R_{la}) == memory capacity of DQN **then**

Sample a random mini-batch n transitions from R_{la}

Set $y_i = r_i^{\mathrm{la}} + \gamma \max_{a^{\mathrm{la}}} Q'_{\mathrm{la}}(s_{i+1}^{\mathrm{la}}, a^{\mathrm{la}}|\theta^{Q'_{\mathrm{la}}})$

Update the upper level Q network by minimizing the loss

$L = \frac{1}{N}\sum_i \left(y_i - Q_{\mathrm{la}}(s_i^{\mathrm{la}}, a_i^{\mathrm{la}}|\theta^{Q_{\mathrm{la}}})\right)^2$

Update the target networks $\theta^{Q'_{\mathrm{la}}} \leftarrow \tau\theta^{Q_{\mathrm{la}}} + (1-\tau)\theta^{Q'_{\mathrm{la}}}$

end if

end for

end for

4. Numerical experiment

As shown in Fig. 4, a corridor that consists of five intersections is built in SUMO (Lopez et al., 2018) to test the HDQPG algorithm. Different distances between adjacent intersections are generated to test the performance of the algorithm. In this section, we first set the number of lanes of each road as 1 (i.e., no lane-changing or by-passing) to test the single lane DDPG algorithm, by which we can show the longitudinal Eco-Driving strategies more clearly. Then, the number of lanes is increased to 3 and the HDQPG is tested. We compare the proposed HDQPG with two existing methods:

1. The default car-following (i.e., Krauss model Krauß et al., 1997; Krauß, 1998) and lane-changing model (i.e., LC2013, developed by Erdmann, 2014) used in SUMO. This is used as the baseline case.
2. The DQN based RL method, which is widely used in existing studies considering Eco-Driving (Shi et al., 2018; Gamage and Lee, 2016, 2017; Hao et al., 2020). The discretized longitudinal acceleration/deceleration, along with three discrete lane-changing indicators, are used as actions. This is also the third action space modeling approach discussed in Section 2.2. One drawback of DQN is that, as the action space increases (i.e., the discretization interval is small), the algorithm will suffer from instability and even fail to learn the optimal actions. However, if one reduces the dimension of action space by introducing larger discretization intervals, the difference between real-world operations (i.e., drivers can choose to apply arbitrary acceleration/deceleration) and the model output (i.e., DQN can only choose from limited discrete acceleration/deceleration) will increase. Based on our numerical experiments with discretization interval varying from 0.2 m/s^2 to 2 m/s^2 with a 0.2 m/s^2 step, the discretization interval is set to 1 m/s^2. The range of the longitudinal acceleration/deceleration is $[-5\ \mathrm{m/s^2}, 3\ \mathrm{m/s^2}]$. Therefore, the action space is $\{-5\ \mathrm{m/s^2}, -4\ \mathrm{m/s^2}, -3\ \mathrm{m/s^2}, -2\ \mathrm{m/s^2}, -1\ \mathrm{m/s^2}, 0\ \mathrm{m/s^2}, 1\ \mathrm{m/s^2}, 2\ \mathrm{m/s^2}, 3\ \mathrm{m/s^2}, -1, 0, 1\}$, where the first nine are for discretized longitudinal accelerations/decelerations, and the last three are for lane-changing indicators (i.e., -1 for changing to the left lane, 0 for stay at current lane, 1 for changing to the right lane). The reward function of the DQN is defined as the sum of the longitudinal (i.e., Eq. (3)) and lateral (i.e., Eq. (9)) rewards: $r^{\mathrm{dqn}} = -w_f g_t + l_t - w_s \max\{0, \tilde{v}_t^e - v_{\mathrm{safe}}^e(t)\} - w_l \times \mathbf{1}_A(x_t)$.

Together with the proposed HDQPG, we first test and compare the three models for the single-lane scenario. As shown in Section 4.2 later, when testing the proposed algorithms for the multi-lane scenario, we also add another model (i.e. the fourth model) for comparison purposes.

4.1. Single lane eco-driving

For the DDPG algorithm used to learn single lane Eco-Driving strategies, we use two $N_i \times 128 \times 64 \times 1$ deep neural networks with fully connected layers to approximate the actor and critic functions, where N_i is the dimension of the state of actor and critic, respectively. For the actor network, the activation function of the output layer is set as tanh such that the output value is continuous in $[-1, 1]$, which is then transformed to the acceleration value by a linear mapping function based on the bounds of accelerations. The learning rate of the actor is set as 0.001 and that of the critic is set as 0.002 based on experiences and our pre-experiments. After trying different discount ratios from 0.8 to 0.99, we set the discount ratio $\gamma = 0.95$. Similarly, different memory capacity values (from 5000 to 15000 with 1000 an increment) are tested, which is set as 10000 finally. The soft replacement parameter is set as $\tau = 0.001$ and the batch size is set as 64.

The volume of this corridor is set as 400 veh/h. The ego vehicle is sampled after first 300 s to guarantee that it is not the first departed vehicle in current episode. When the ego vehicle exits the corridor, the current episode terminates. We train the DDPG algorithm for 1000 episodes, each of which uses different random seeds. For every 10 training episodes, the learned strategies are tested by 5 testing episodes. All settings (i.e., topology of the intersection, traffic volume, etc.) of these five test episodes are the same except that different random seeds are used to generate vehicles. Thus, the ego vehicle may enter the simulation network at different times with different surrounding environments (see Section 4.1.2). The baseline case is generated by running the SUMO simulation

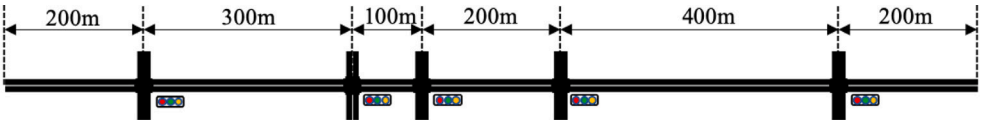

Fig. 4. Test Scenario.

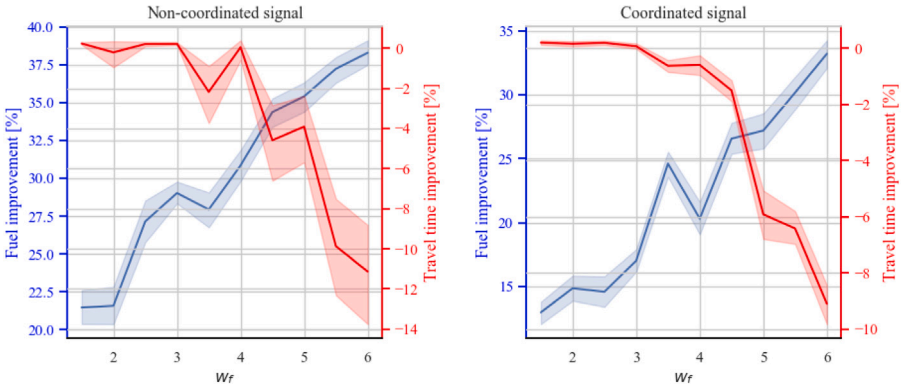

Fig. 5. Fuel and travel time improvements under different w_f for non-coordinated and coordinated signal.

without any external control, i.e., the ego vehicle is controlled by SUMO's default car-following model. We test the algorithms under two signal settings: (1) non-coordinated signal plan where the phases and offsets along the corridor are not optimized; and (2) coordinated signal plan where the phases and offsets are optimized based on free flow travel times between adjacent intersections to generate green waves.

4.1.1. Parameter tuning and general performance

Since we consider both the fuel consumption and travel time in the reward function, the performance of the RL controller might be quite different under different weights of the two. According to Eq. (3), intuitively, increasing w_f should improve the fuel performance but might increase the travel time. Although we focus on Eco-Driving, the travel time should not be sacrificed too much. In order to balance these two objectives, we fix the parameter $w_s = 1$ and train the DDPG algorithm under different w_f. The results are shown in Fig. 5. The solid lines in the figure represent the average values and the shaded area around each solid line indicates the 95% confidence interval.

Under the non-coordinated signal setting, as w_f increases from 1.5 to 4, the performance of fuel consumption improves from 22% to 31% while the travel time maintains at a constant level as the baseline case. As w_f keeps increasing, although the fuel consumption could be further reduced, the travel time would increase significantly. If $w_f < 1.5$, the fuel consumption improvements will be compromised too much. The same trend can also be found under the coordinated signal setting: the fuel-saving increases from 13% to 27% when w_f increases from 1.5 to 4.5, while travel time increment is limited under 2%. Therefore, we choose $w_f = 4.5$ as the best balance between the performances of fuel and travel time, since the performance of fuel consumption can be improved significantly and the increment of travel time can be limited under 5% for both signal settings. A same process for the DQN based Eco-Driving is conducted, and $w_f = 5$ is chosen. Notice that later in this section when we compare the performances of the proposed algorithm with the DQN-based Eco-Driving method, we show fuel consumption and travel times separately. Therefore, setting different w_f parameters for the two algorithms are acceptable.

Fig. 6 shows the reward (with the weight parameters $w_f = 4.5, w_s = 1.0$), fuel and travel time improvements of the DDPG single lane Eco-Driving algorithm compared with the baseline controller during the training process under the coordinated signal setting (the same trend is found under the non-coordinated signal setting and is omitted here for brevity). It is shown that the reward decreases at the starting period, but then increases dramatically in less than 50 episodes. After that, the reward increases slowly with fluctuations and finally stabilizes around 5. The fuel consumption improvement shows a similar trend. The fuel improvement decreases dramatically at the early training period (i.e., the fuel consumption increases dramatically), but then increases over 0 and finally stabilize around 27%. Meanwhile, the travel time becomes a little worse, but is no more than 5% longer compared with the baseline controller.

Table 2 shows the performances of the three control algorithms. It is shown that, under either the coordinated or the non-coordinated signal setting (five test cases for each setting), both the DQN based and DDPG based RL controllers can improve the fuel performance compared with the baseline case. Meanwhile, DDPG significantly outperforms DQN in terms of reducing fuel consumption. In terms of travel times, DQN outperforms DDPG very mildly under the coordinated signal setting. However, under the non-coordinated signal setting, DQN is unstable with large deviations (see Section 4.1.2 for more explanations), while DDPG can maintain almost the same travel time performance as the baseline case. We then increase the test cases to 100 to check if the

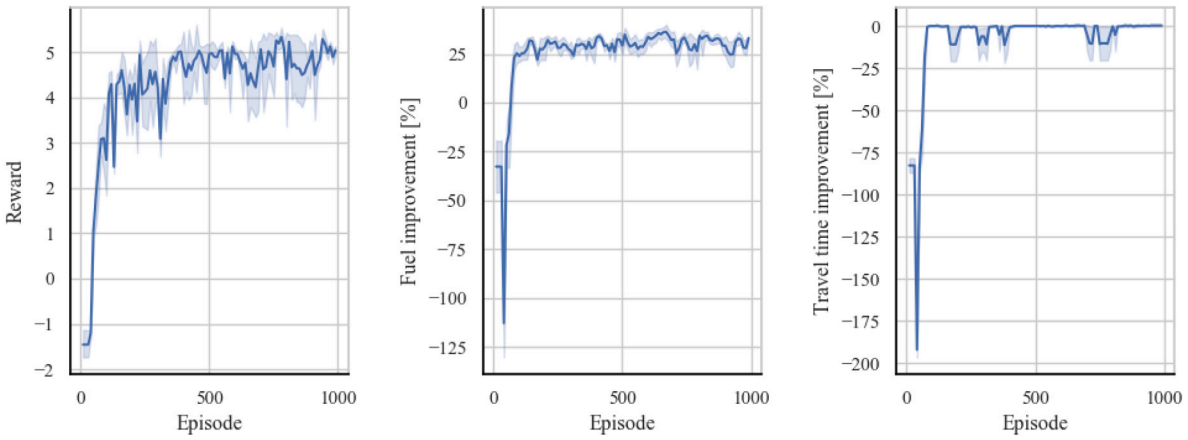

Fig. 6. Performance of DDPG-based single lane Eco-Driving under coordinated signal.

Table 2
Performance of different single lane control algorithms under two signal settings (5 test cases).

		Fuel consumption [ml]					Travel time [s]				
		Base	DQN	Imp.	DDPG	Imp.	Base	DQN	Imp.	DDPG	Imp.
Coord.	Case 1	133.40	111.57	16.36%	95.54	28.38%	95	95	0.00%	100	−5.26%
	Case 2	135.94	122.98	9.53%	93.15	31.48%	99	99	0.00%	103	−4.04%
	Case 3	184.46	162.07	12.14%	116.56	36.81%	135	135	0.00%	140	−3.70%
	Case 4	132.37	120.71	8.81%	88.77	32.94%	98	98	0.00%	104	−6.12%
	Case 5	170.16	148.41	12.79%	109.52	35.64%	131	131	0.00%	135	−3.05%
Non-coord.	Case 1	185.48	158.52	14.53%	134.78	27.34%	151	151	0.00%	151	0.00%
	Case 2	198.12	170.86	13.76%	131.66	33.55%	155	155	0.00%	155	0.00%
	Case 3	216.99	214.03	1.36%	144.39	33.46%	155	194	−25.16%	154	0.65%
	Case 4	182.72	169.40	7.29%	118.92	34.91%	152	152	0.00%	152	0.00%
	Case 5	217.36	215.23	0.98%	134.25	38.24%	150	189	−26.00%	149	0.67%

Table 3
Performance of different single lane control algorithms under two signal settings (summary of 100 test cases).

		Fuel consumption [ml]			Travel time [s]		
		Base	DQN	DDPG	Base	DQN	DDPG
Coord.	Avg	150.46	147.57	112.82	110.82	116.29	112.32
	Std	21.61	24.35	19.13	17.81	18.81	16.10
Non-coord.	Avg	199.76	214.81	138.30	156.04	167.77	162.68
	Std	24.11	32.57	19.13	16.58	22.87	20.99

above findings are consistent among different test cases. The results are shown in Table 3. We can see that with more test cases, DDPG actually outperforms DQN: the mean and standard deviation of either fuel consumption or travel time are smaller for DDPG for both the coordinated and coordinated settings. DDPG also outperforms the baseline algorithm in terms of fuel consumption with slight degradation (less than 5%) when travel time is considered.

4.1.2. Analysis of the eco-driving mechanism

Fig. 7 shows the spatial–temporal diagrams of the ego vehicle for the first four test cases in Table 2 under the non-coordinated signal setting (Case 5 shows the same pattern as Case 3, and is omitted here for brevity). The only difference among the four figures is that they use different random seeds, i.e., the times that the ego vehicle entering the corridor are different. For all three control methods, under the same case, all components of the simulation including the IDs and entering times of the ego vehicle and its front vehicle are the same. In the figure, the horizontal red lines indicate the red times of the signals. We can see that the key difference of the driving strategies between the baseline and the two RL-based control methods is that, if the ego vehicle cannot pass the intersection in current green phase, RL-based Eco-Driving methods will slow down the ego vehicle earlier compared with the baseline control method. To illustrate this more clearly, we show the spatial–temporal diagram, speed, and accumulated fuel consumption of test case 2 in Fig. 8. Under the baseline algorithm, the ego vehicle maintains a high speed to approach the intersection, then decelerate to zero within a relatively short time interval. Under both DQN and DDPG, the ego vehicle starts to decelerate earlier than the baseline case. Since fuel consumption rate is zero when decelerating, the two RL controllers could save fuel through this strategy. Compared with DQN, the speed profile learned by DDPG is smoother without large fluctuations. DQN can

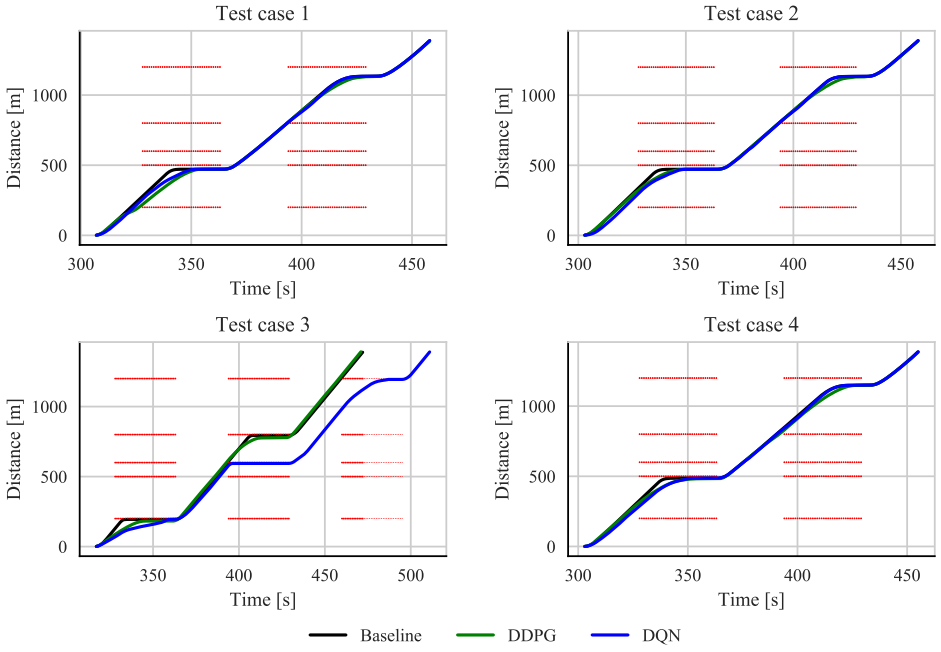

Fig. 7. Spatial–temporal diagrams of the ego vehicle under non-coordinated signal setting.

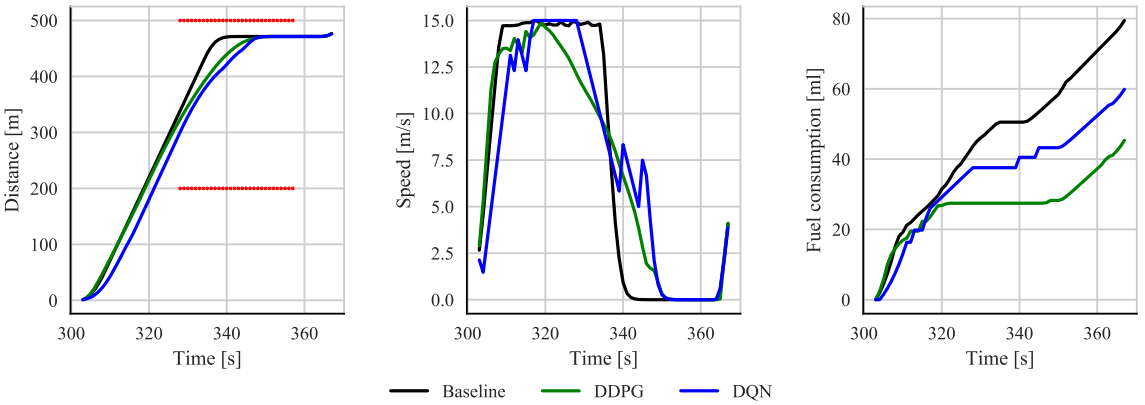

Fig. 8. Spatial–temporal diagram, speed, and accumulated fuel of test case 2 under non-coordinated signal setting.

only generate discrete acceleration/deceleration values, which will make the speed profile less smooth if the optimal actions are not perfectly learned. Such fluctuations will not only increase the fuel consumption, but also introduce delays that can be amplified by traffic signals; see test case 3 in Fig. 7.

In Figs. 7 and 8, signal plans of the five intersections on this corridor are not coordinated. The ego vehicle encounters multiple stop-and-goes under every test case. We then analyze the coordinated signal setting. Fig. 9 shows the spatial–temporal diagram, speed and accumulated fuel consumption of test case 2 under the coordinated signal setting. It shows that the ego vehicle under the baseline control first accelerates to a high speed and maintains that speed with slight fluctuations during the whole simulation period. Under DQN, the ego vehicle behaves in the same way. The speed profile under DQN is a straight line during most of the cruising period since DQN can only generate discrete accelerations with an interval of 1 m/s^2. Such operations can reduce the fuel consumption caused by the speed fluctuations. Under DDPG, the ego vehicle also first accelerates to a high speed, then periodically "pulse and glide" (PnG) around a high speed. This PnG operation has been proven to be fuel-efficient due to the non-linearity of fuel consumption characteristics of engines (Li et al., 2012, 2016; Xu et al., 2015), especially the fact that fuel consumption during deceleration is 0. This phenomenon becomes more and more clear as the fuel weight w_f increases. Under low w_f, the ego vehicle cares more about travel time, which encourages high speed. Under higher w_f, saving fuel is given a higher priority, for which the ego vehicle learns the PnG strategy. In addition, it is found that the larger the amplitude of the speed fluctuation, the lower the average speed (i.e., the longer the travel time), while more fuel-savings can be achieved at the same time. PnG developed

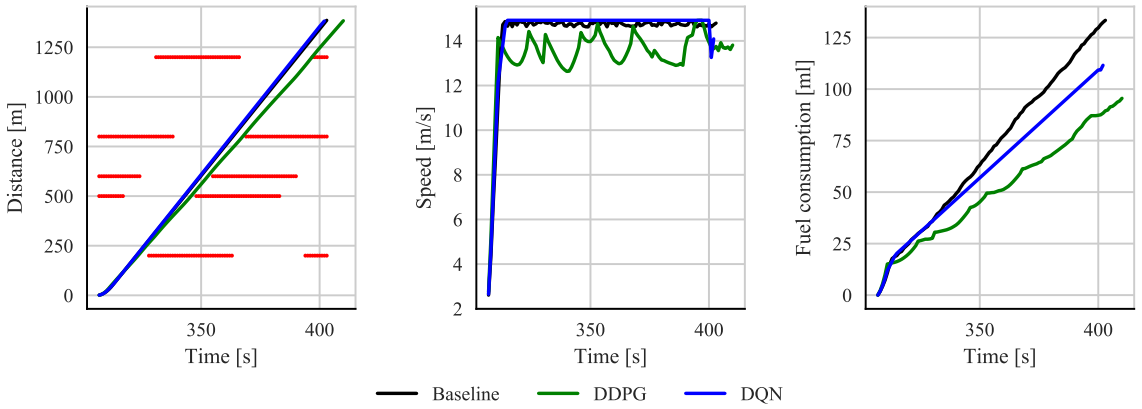

Fig. 9. Spatial–temporal diagram, speed, and accumulated fuel of test case 2 under coordinated signal setting.

in literature (e.g., Li et al., 2012, 2016; Xu et al., 2015) requires sophisticated models and are usually applied to highways. Our proposed RL controller is purely data driven, and can properly learn this PnG strategy on urban streets. Also the fact that we model vehicle acceleration/deceleration as continuous variables is the main reason why the proposed DDPG method can properly learn the PnG strategy. Since existing DQN-based RL methods discretize the acceleration/deceleration space, it is much harder for these methods to properly learn PnG.

In summary, there are essentially two strategies learned by the DDPG to save fuel: (1) decelerate earlier before the traffic signal if the ego vehicle cannot pass the intersection in current green phase, and (2) learn and apply the PnG method to reduce fuel consumption while cruising and following the leading vehicle (if there is one). These two strategies are consistent with the fuel-saving operations found by model-based methods as well as learning-based methods in the literature. Previous studies usually explored these two strategies separately using different methods and under specific (such as highway) scenarios, while the proposed DDPG-based method can properly learn and integrate these two strategies in one pure data-driven framework.

4.2. Multiple-lane eco-driving

The experiment settings of multi-lane Eco-Driving are similar to those used in the single-lane experiment described above, with the extension that each road in Fig. 4 is now three lanes. The total volume is tripled to 1200 veh/h. Each vehicle enters the corridor from the "best lane" defined by SUMO, and can change lanes based on traffic conditions. In order to evaluate the control performance under unusual traffic conditions, we generate two such cases at the second and fifth road segments (i.e., the two longest segments as shown in Fig. 4). When the ego vehicle drives along one of those two segments, if the speed of the ego vehicle is larger than 10 m/s and there is a front vehicle on the same lane with speed larger than 10 m/s, the front vehicle will stay on the current lane and decelerate to 2 m/s within 4 s and maintain this speed until exiting the corridor. These "purposely" generated scenarios are used to mimic unusual traffic conditions, such as lane drops or traffic accidents that cause the leading vehicles to slow down abruptly. The way how these unusual conditions are generated also imply that for different test cases in this subsection, the locations where this occurs are likely different, instead of fixed at a pre-defined location. This can help test if the algorithms can deal with unusual conditions in general.

An $17 \times 128 \times 3$ neural network with fully connected layers is used to approximate the lateral Q function. The HDQPG Eco-Driving algorithm is tested with the parameter $w_f = 4.5$ and $w_l = 15$, and the DQN-based Eco-Driving algorithm is tested with the parameter $w_f = 5.0$ and $w_l = 15$ after fine-tuning. Both HDQPG and DQN are trained for 1000 episodes. In order to show the impact of the lateral RL controller, we also test the fourth method: the longitudinal acceleration/deceleration is generated from DDPG and the lateral lane-changing operation is controlled by the SUMO default model (Erdmann, 2015). We refer this method as BDDPG (base lateral plus DDPG longitudinal). That is, the proposed HDQPG method and BDDPG differ only in the lateral lane changing control, which can help illustrate the performance of the proposed RL-based lateral control model. Note that we do not test another combination, i.e., SUMO for longitudinal control and RL for lateral control, since the lateral RL control is based on some input (i.e., the critic's value) from the longitudinal RL control.

4.2.1. Comparison of the overall performance

The detailed results of five test cases are shown in Table 4 and the summary of 100 test cases are shown in Table 5. The first observation from Table 4 is that HDQPG, BDDPG and DQN can all improve the performance under the coordinated signal setting compared with the baseline controller. HDQPG outperforms DQN regarding fuel consumption, but performs slightly worse when it comes to travel time for the five cases. However, under the non-coordinated signal setting, DQN fails to improve the driving performance for four out of the total five test cases: the fuel consumption and travel time of DQN are much worse than the baseline controller. It is hard for DQN to converge under such multi-lane non-coordinated signal settings, because compared with coordinated

Table 4
Performance of different multi-lane control algorithms under two signal settings (detail of 5 test cases).

		Coord.					Non-coord.				
		Case 1	Case 2	Case 3	Case 4	Case 5	Case 1	Case 2	Case 3	Case 4	Case 5
Fuel [ml]	Base	142	141	196	141	131	193	211	237	202	198
	DQN	119	112	154	112	112	190	497	287	346	319
	Imp.	16%	20%	21%	20%	15%	2%	−136%	−21%	−71%	−61%
	BDDPG	117	109	156	110	114	145	152	165	134	135
	Imp.	18%	22%	20%	22%	13%	25%	28%	31%	34%	32%
	HDQPG	98	103	128	99	96	104	119	204	125	107
	Imp.	31%	27%	35%	30%	27%	46%	43%	14%	38%	46%
Time [s]	Base	99	97	150	97	96	149	156	198	154	148
	DQN	98	95	144	95	95	152	567	287	360	307
	Imp.	1%	2%	4%	2%	1%	−2%	−263%	−45%	−134%	−107%
	BDDPG	100	99	155	99	96	151	158	197	154	147
	Imp.	−1%	−2%	−3%	−2%	0%	−1%	−1%	1%	0%	1%
	HDQPG	108	109	154	107	102	148	157	271	153	149
	Imp.	−9%	−12%	−3%	−10%	−6%	1%	−1%	−37%	1%	−1%

Table 5
Performance of different single lane control algorithms under two signal settings (summary of 100 test cases).

		Fuel consumption [ml]				Travel time [s]			
		Base	DQN	BDDPG	HDQPG	Base	DQN	BDDPG	HDQPG
Coord.	Avg	175.42	169.46	136.91	120.05	129.97	154.84	129.11	131.76
	Std	44.71	79.32	32.65	29.15	42.39	87.03	40.90	34.90
Non-coord.	Avg	218.36	254.26	154.80	133.87	168.59	236.93	174.67	179.87
	Std	32.49	65.00	26.54	31.28	32.71	75.10	33.77	39.55

signal settings, the number of possible states (under non-coordinated settings) are much larger. Together with the relatively large action space, DQN suffers from inefficient learning. HDQPG, on the other hand, can improve the performance of fuel consumption by up to 35% and 46%, respectively, under both coordinated and non-coordinated signal settings compared with the baseline controller. One can observe a relatively poor performance of HDQPG under Case 3 in the non-coordinated signal setting, which takes 37% longer time to finish the trip. This is due to the fact that, once the ego vehicle is stopped by a red signal, it has to wait until the signal turns to green. A single stop will make the travel time much longer since the total travel distance of the studied scenario is only 1400 m. We believe that such increment of travel time should be reasonable, especially when fuel consumption can still be reduced by 14%. In addition, under the multi-lane scenario, since there is only one ego vehicle, the following vehicles could change lane to bypass the ego vehicle if they feel that the ego vehicle is too slow. Therefore, the travel time increment only holds for the ego vehicle, while other vehicles usually do not suffer long extra travel times. With 100 test cases in Table 5, similar results can be observed. In particular, HDQPG outperforms the other three algorithms in terms of fuel consumption with much smaller average and smaller (or comparable) standard deviation. For travel times, HDQPG outperforms DQN but is slightly worse than the baseline algorithm and BDDPG.

4.2.2. Lateral operation comparison of HDQPG and BDDPG

In order to further demonstrate how the lateral RL controller improves the performance, we next focus on HDQPG and BDDPG. For BDDPG, the lateral lane-changing behavior is controlled by the default model of SUMO, while the longitudinal acceleration/deceleration is generated by the single lane DDPG. Table 4 shows that both HDQPG and BDDPG can improve the performance of the baseline controller under either the coordinated or the non-coordinated signal setting. Note that in the baseline controller, the ego vehicle can also change lanes to avoid congestion when the front vehicle "purposely" decelerates. The travel times of both HDQPG and BDDPG are similar to those of the baseline controller (except Case 3 under non-coordinated signal, due to the reason discussed above), under either the coordinated or non-coordinated signal setting. In terms of fuel consumption, under both coordinated and non-coordinated signals, HDQPG performs significantly better than BDDPG. Such difference between BDDPG and HDQPG are mainly resulted from the different objectives of these two lateral controllers: the SUMO lane-changing model of BDDPG aims to improve the speed gain (Erdmann, 2015), while HDQPG tries to reduce both travel time and fuel consumption.

It is hard to compare in detail the differences between the lane-changing operations of HDQPG and BDDPG. One reason lies in the difficulty of capturing a scenario where all surrounding environments of the ego vehicle under HDQPG and BDDPG are exactly the same. In addition, since HDQPG considers the long-term fuel performance, the lane-changing decision at a specific time might not be the best for just that particular time, which however may help achieve the greater long-term benefit. Here we show a scenario where the surrounding environments of the ego vehicle under HDQPG and BDDPG are almost the same. We qualitatively compare the lane-changing operation differences between HDQPG and BDDPG. Fig. 10 shows such a scenario, which happens on the fifth road segment in Case 2 under the non-coordinated signal setting. As mentioned earlier, on this segment, the ego vehicle will encounter a manually created "slow down", i.e., the front vehicle abruptly decreases to a very low speed (2 m/s) and maintains that speed until

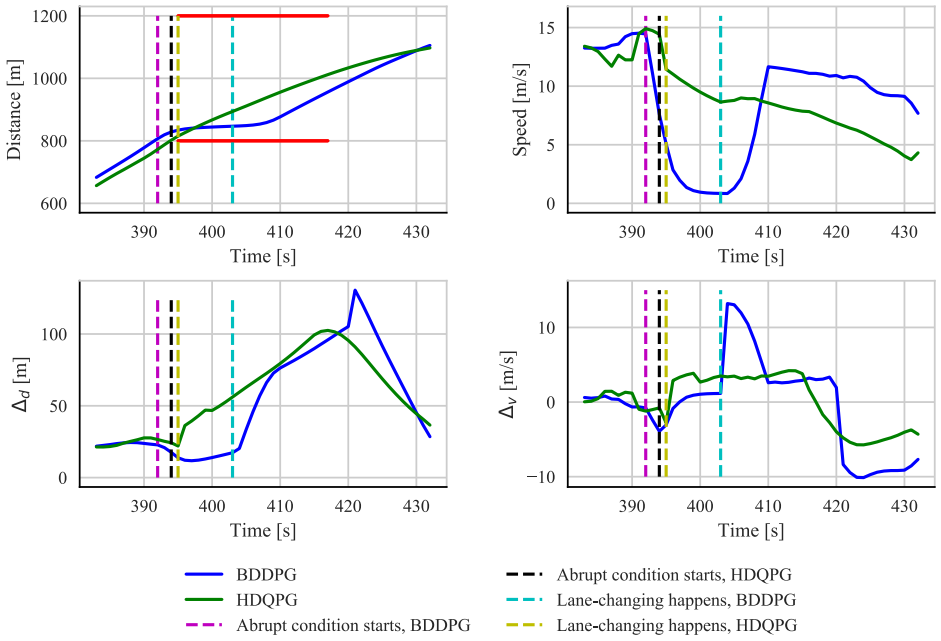

Fig. 10. Lane-changing behavior of BDDPG and HDQPG. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the end of the simulation. In Fig. 10, the red solid horizontal lines in the upper left plot are used to indicate the duration of the red signal phase, the dashed cyan and black vertical lines show when the front vehicle starts to abruptly decelerate under BDDPG and HDQPG respectively, and the dashed magenta and yellow lines show when the ego vehicle changes lane under BDDPG and HDQPG respectively. The solid blue and green lines show, under BDDPG and HDQPG respectively, the distance (upper left plot) and speed (upper right plot) of the ego-vehicle, and the distance to the front vehicle (lower left plot) and the speed difference between the front vehicle and the ego vehicle (lower right plot). Note that the trajectories of the ego vehicle under either BDDPG or HDQPG are aligned together in terms of longitudinal travel distances (which might be on different lanes).

We can observe that, the abrupt deceleration happens at different times for BDDPG and HDQPG. This is because, for BDDPG and HDQPG, the control sequences are different before the abrupt deceleration occurs, which results in different driving scenarios when they arrive at the road segment and location where the abrupt deceleration happens. Such different scenarios will trigger the abrupt deceleration at different times based on the logic discussed in Section 4.2. Before this abrupt condition occurs, the ego vehicles under BDDPG and HDQPG drives at a similar environment, i.e., each of them follows the front vehicle with similar speed difference as well as similar distance, which can be shown from the lower two plots in Fig. 10. When the front vehicle starts to abruptly decelerate, the ego vehicle under HDQPG changes lane right after the abrupt deceleration starts, which enables it to maintain a relatively high speed to gain more travel time and fuel consumption benefits. Under BDDPG, the ego vehicle first slows down sharply to a low speed, following the front vehicle; and then after about 11 s, the ego vehicle changes lane to gain more speed benefit. The fuel improvements of BDDPG and HDQPG are 28% and 43% respectively compared with the baseline controller, while the travel time improvements of BDDPG and HDQPG are both around −1%. There are two reasons that may help explain the relatively long delay (i.e. 11 s) before the ego vehicle makes the lane changing under BDDPG. First, the tactical lane-changing model used in SUMO uses the accumulative speed gain probability to prevent oscillations. Thus, it takes time (11 s in this test case) to accumulate the speed gain probability to reach the pre-defined threshold (see Erdmann, 2015 for details). Second, we cannot guarantee the traffic conditions of adjacent lanes are exactly the same for BDDPG and HDQPG. BDDPG may be prohibited to change lane due to traffic conditions of adjacent lanes even the speed gain probability has reached the threshold. The scenario shown in Fig. 10 is part of Case 2 under the non-coordinated signal setting (see Table 4). For other cases, the patterns shown in Fig. 10 are common, although it is possible that both BDDPG and HDQPG may decelerate to low speeds first since it is not safe to change lane immediately.

4.2.3. Sensitivity analysis of the proposed HDQPG algorithm

We next test the sensitivity of the HDQPG algorithm under different traffic volumes. Apart from the volume used in the aforementioned analyses, i.e., 400 veh/(lane.h), we test four additional volumes [200, 600, 800, 1000] veh/(lane.h). Since the DQN method fails under most non-coordinated signal settings, we only compare the performances of the baseline controller, BDDPG, and HDQPG. We also set the number of test cases to 30, each of which has a different random seed. Fig. 11 shows the average fuel consumption of the baseline controller, BDDPG, and HDQPG under the five different traffic volumes and different traffic signal settings. Under both non-coordinated and coordinated signal settings, the fuel consumption of HDQPG and BDDPG follows the

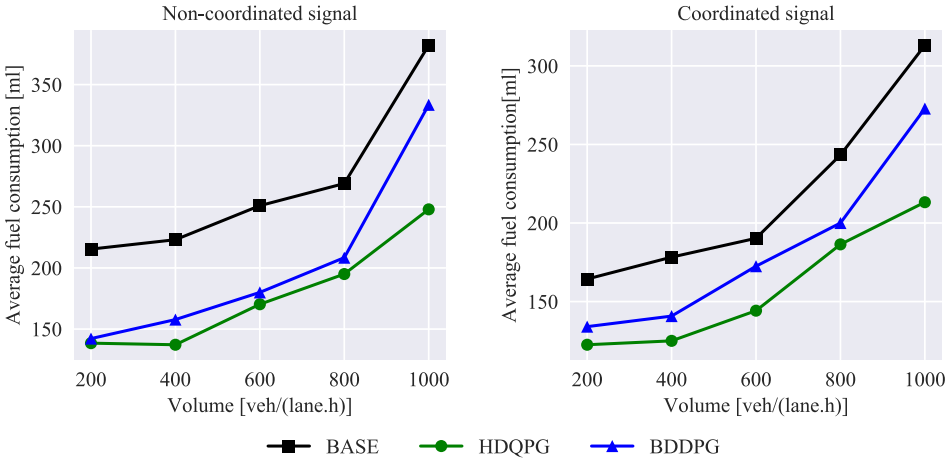

Fig. 11. Fuel consumption of different methods under different volumes.

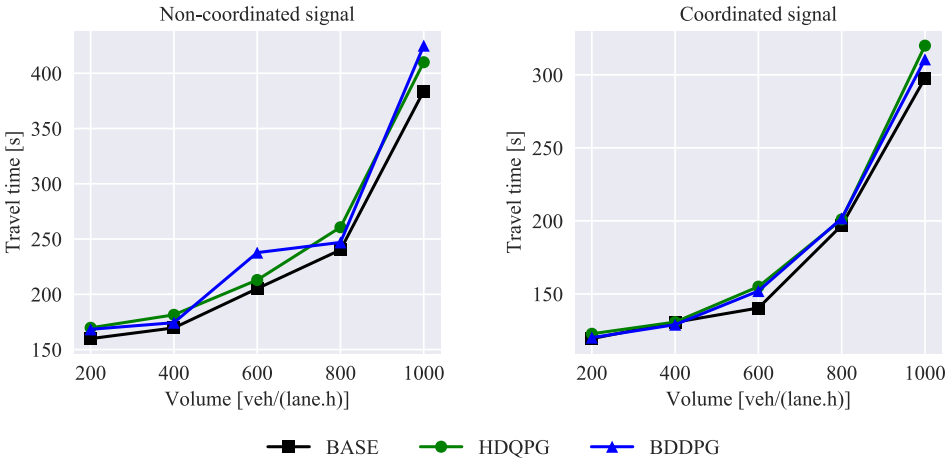

Fig. 12. Travel time of different methods under different volumes.

same trend, albeit with significant improvements, compared with that of the baseline controller. Under both settings, the fuel consumption of completing the corridor trip increases as the traffic volume increases. Higher volume causes heavier congestion, which also increases fuel consumption. Under the same volume, the fuel consumption of the same controller under the coordinated setting is generally lower than that of the non-coordinated setting, which is consistent with the goal of signal coordination along the corridor. More specifically, HDQPG outperforms BDDPG for all the five demand scenarios due to the reasons discussed above. Meanwhile, as shown in Fig. 12, the travel times of these three methods are very close. This shows that HDQPG is quite robust under different traffic volumes.

It should be noted that we did not compare the proposed algorithm with traditional model-based Eco-Driving methods directly. This is because the driving scenario considered in this paper (a multi-lane urban signalized corridor) is too complex to apply (a single) model-based Eco-Driving methods, which is also one of the main motivations to develop RL-based methods. Traditional model-based methods usually simplify the driving scenario and focus on one specific task (e.g., car-following on highways Li et al., 2012 or intersection passing Rakha and Kamalanathsharma, 2011, etc.). To the best of our knowledge, currently, there are no model-based Eco-Driving methods that can deal with both longitudinal and lateral operations on a multi-lane signalized urban road. Therefore, we do not compare the proposed method with existing model-based methods. Nevertheless, we can compare the overall performance of the proposed method and model-based methods in an indirect way. Since both the proposed method and most model-based methods in the literature are compared with some baseline controller (e.g., IDM-model, Krauss model, etc.), we can compare the performance improvements. Although this is not very accurate since the driving scenarios of the proposed method and other model-based methods are quite different, we can still get some useful insights. For example, the reported fuel improvement in the literature is from 3.5%–60% for specific driving tasks under model-based Eco-Driving methods. The proposed method yields 16%–46% fuel improvement. Although the maximum gain is less than the extreme case found in the literature (which is achieved by the PnG technique under the car-following scenario Li et al., 2016), this still shows that the proposed method performs comparably well under the more-complex multi-lane urban driving scenarios.

5. Conclusion

In this paper, we developed a hybrid deep Q-learning and policy gradient (HDQPG) based Eco-Driving model for vehicles driving along signalized corridors under the environment of connected and automated vehicles (CAVs). Both longitudinal acceleration/deceleration and lateral lane-changing operations were considered. The information of surrounding vehicles and upcoming traffic signals were collected by on-board sensors, V2V, and V2I techniques. We also designed specific states, actions, and reward functions for the longitudinal and lateral RL models respectively, and developed a "checking-feedback-learning" (CFL) framework to ensure driving safety and decision consistency by encouraging the RL algorithms to learn safe driving behaviors and consistent decisions.

A deep deterministic policy gradient algorithm was designed to learn the longitudinal operations and a deep Q-learning neural network was developed to make the lane-changing decisions. We tested the proposed HDQPG model on a three-lane five-intersection corridor with different traffic conditions. The results showed that HDQPG outperforms existing RL-based Eco-Driving methods, and could learn two basic fuel-saving strategies (i.e., decelerating earlier if the ego vehicle cannot pass the upcoming intersection in current green phase; and using pulse and glide (PnG) operation to make efficient use of kinetic energy when cruising), which have been proven to be fuel-efficient by model-based methods in the literature. The HDQPG algorithm could also deal with unusual driving conditions like abrupt deceleration of the front vehicle by changing lanes at proper times. Compared with the baseline control method, under both coordinated and non-coordinated signal settings, the fuel performance could be improved by 27%–35% and 14%–46%, respectively.

There are several limitations of the proposed HDQPG model, which merits further investigations. First, we only test the proposed model and algorithm on a traffic corridor in simulation, which needs to be further tested using more case studies and ideally on real-world traffic corridors or test beds. In order to implement the proposed algorithm in real-world, a lower level regulator might be needed to smooth the acceleration profile generated by the RL controller and generate the final commands to the actuators (e.g., throttle angle of the engine and angles of the braking/acceleration pedal). Second, we only use the data from five surrounding vehicles of the ego-vehicle to design the RL algorithms. Using data of more vehicles to design more efficient and robust Eco-Driving algorithms is an interesting future research direction once the V2V and V2X techniques become more widely deployed. Third, we did not consider the routing problem in this paper since the focus is on signalized corridors. Designing RL-based algorithms for a transportation network in which the ego vehicle can learn and find the best routes to follow and then the best lanes for left-turn/right-turn/go-straight when approaching an intersection is an interesting and challenging topic. Fourth, the fuel consumption model in SUMO is based on the 3rd degree polynomials over speed and acceleration (Krajzewicz et al., 2015). Although this polynomial function was fit to detailed engine models based on the data of Handbook Emission Factors for Road Transport (HBEFA) (NHTSA, 2020), it still may not be very accurate, especially when the lateral operations are considered. Integrating a more accurate fuel consumption model as well as a more accurate vehicle dynamics model is needed in future research. Fifth, we developed the CFL ("checking-feedback-learning") framework to ensure safe (consistent) driving behaviors (decisions) of the actions generated by the RL algorithms. It is an interesting topic in the future to investigate more sophisticated RL methods that can directly guarantee safe behaviors and consistent decisions, without applying the CFL framework. Sixth, we focused on the control of only one vehicle (i.e. the ego vehicle that is a CAV) in this paper. This is appropriate if the penetration of CAVs are very small. When the penetration of CAVs increases, it will be interesting to study the coupled signal control with Eco-Driving as an integrated problem to simultaneously optimize traffic signal timings and the movements/operations of all CAVs on a corridor or even on a network. Solving this integrative problem may lead to more congestion reduction and fuel saving benefits, as indicated in Guo et al. (2019). The authors may investigate the above-identified topics in future research and results may be presented in subsequent papers.

CRediT authorship contribution statement

Qiangqiang Guo: Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Roles/Writing - original draft, Writing - review & editing. **Ohay Angah:** Data curation, Investigation, Validation, Software. **Zhijun Liu:** Data curation, Investigation, Validation. **Xuegang (Jeff) Ban:** Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Roles/Writing, Writing - review & editing.

Acknowledgments

The authors thank the three anonymous reviewers for their constructive comments that helped significantly improve the qualify of earlier versions of the manuscript. This research is partially supported by the Connected Cities with Smart Transportation (C2Smart) University of Transportation Center at the New York University via a grant to the University of Washington.

All authors reviewed the results and approved the final version of the manuscript.

References

Barić, D., Zovak, G., Periša, M., 2013. Effects of eco-drive education on the reduction of fuel consumption and CO2 emissions. Promet-Traffic Transp. 25 (3), 265–272.

Borken, J., Steller, H., Meretei, T., Vanhove, F., 2007. Global and country inventory of road passenger and freight transportation: Fuel consumption and emissions of air pollutants in year 2000. Transp. Res. Rec. 2011 (1), 127–136.

Erdmann, J., 2014. Lane-changing model in SUMO. In: Proceedings of the SUMO2014 Modeling Mobility with Open Data, 24. Deutsches Zentrum für Luft-und Raumfahrt eV, pp. 77–88.
Erdmann, J., 2015. SUMO's lane-changing model. In: Modeling Mobility with Open Data. Springer, pp. 105–123.
Gamage, H.D., Lee, J., 2016. Machine learning approach for self-learning eco-speed control. In: Young, W. (Ed.), Australasian Transport Research Forum 2016 Proceedings. Australasian Transport Research Forum, https://www.australasiantransportresearchforum.org.au/papers/2016.
Gamage, H.D., Lee, J.B., 2017. Reinforcement learning based driving speed control for two vehicle scenario, In: Australasian Transport Research Forum, ATRF, 39th.
Guo, Q., Li, L., Ban, X.J., 2019. Urban traffic signal control with connected and automated vehicles: A survey. Transp. Res. C 101, 313–334.
Hao, P., Wei, Z., Bai, Z., Barth, M.J., 2020. Developing an adaptive strategy for connected eco-driving under uncertain traffic and signal conditions. In: UC Davis: National Center for Sustainable Transportation. UC Davis, http://dx.doi.org/10.7922/G2F18WZ1.
Hellström, E., Ivarsson, M., Åslund, J., Nielsen, L., 2009. Look-ahead control for heavy trucks to minimize trip time and fuel consumption. Control Eng. Pract. 17 (2), 245–254.
Hu, Y., Li, W., Xu, K., Zahid, T., Qin, F., Li, C., 2018. Energy management strategy for a hybrid electric vehicle based on deep reinforcement learning. Appl. Sci. 8 (2), 187.
Huang, Y., Ng, E.C., Zhou, J.L., Surawski, N.C., Chan, E.F., Hong, G., 2018. Eco-driving technology for sustainable road transport: A review. Renew. Sustain. Energy Rev. 93, 596–609.
Krajzewicz, D., Behrisch, M., Wagner, P., Luz, R., Krumnow, M., 2015. Second generation of pollutant emission models for SUMO. In: Modeling Mobility with Open Data. Springer, pp. 203–221.
Krauß, S., 1998. Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics (Ph.D. thesis).
Krauß, S., Wagner, P., Gawron, C., 1997. Metastable states in a microscopic model of traffic flow. Phys. Rev. E 55 (5), 5597.
Li, S.E., Guo, Q., Xin, L., Cheng, B., Li, K., 2016. Fuel-saving servo-loop control for an adaptive cruise control system of road vehicles with step-gear transmission. IEEE Trans. Veh. Technol. 66 (3), 2033–2043.
Li, S.E., Peng, H., 2012. Strategies to minimize the fuel consumption of passenger cars during car-following scenarios. Proc. Inst. Mech. Eng. D 226 (3), 419–429.
Li, S.E., Peng, H., Li, K., Wang, J., 2012. Minimum fuel control strategy in automated car-following scenarios. IEEE Trans. Veh. Technol. 61 (3), 998–1007.
Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D., 2015. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.
Lin, L.-J., 1992. Self-improving reactive agents based on reinforcement learning, planning and teaching. Mach. Learn. 8 (3–4), 293–321.
Lopez, P.A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., Wießner, E., 2018. Microscopic traffic simulation using SUMO. In: IEEE Intelligent Transportation Systems Conference, ITSC, The 21st IEEE International Conference on Intelligent Transportation Systems. IEEE, https://elib.dlr.de/124092/.
Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M., 2013. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.
Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533.
NHTSA, 2016. Automated vehicles for safety. https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety/issue-road-self-driving.
NHTSA, 2020. SUMO fuel model data. https://www.hbefa.net/e/index.html.
Qi, X., Luo, Y., Wu, G., Boriboonsomsin, K., Barth, M.J., 2017. Deep reinforcement learning-based vehicle energy efficiency autonomous learning system. In: 2017 IEEE Intelligent Vehicles Symposium, IV. IEEE, pp. 1228–1233.
Qi, X., Luo, Y., Wu, G., Boriboonsomsin, K., Barth, M., 2019. Deep reinforcement learning enabled self-learning control for energy efficient driving. Transp. Res. C 99, 67–81.
Qi, X., Wu, G., Boriboonsomsin, K., Barth, M.J., Gonder, J., 2016. Data-driven reinforcement learning–based real-time energy management system for plug-in hybrid electric vehicles. Transp. Res. Rec. 2572 (1), 1–8.
Qu, X., Yu, Y., Zhou, M., Lin, C.-T., Wang, X., 2020. Jointly dampening traffic oscillations and improving energy consumption with electric, connected and automated vehicles: A reinforcement learning based approach. Appl. Energy 257, 114030.
Rakha, H., Kamalanathsharma, R.K., 2011. Eco-driving at signalized intersections using V2I communication. In: 2011 14th International IEEE Conference on Intelligent Transportation Systems, ITSC. IEEE, pp. 341–346.
Row, S.J., 2010. Intellidrive: Safer, smarter, greener. Public Roads 74 (1).
Shi, J., Qiao, F., Li, Q., Yu, L., Hu, Y., 2018. Application and evaluation of the reinforcement learning approach to eco-driving at intersections under infrastructure-to-vehicle communications. Transp. Res. Rec. 2672 (25), 89–98.
Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M., 2014. Deterministic Policy Gradient Algorithms.
Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: An Introduction. MIT press.
Tsitsiklis, J., Van Roy, B., 1996. An analysis of temporal-difference learning with function approximation. Technical Report LIDS-P-2322, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology.
Walnum, H.J., Simonsen, M., 2015. Does driving behavior matter? An analysis of fuel consumption data from heavy-duty trucks. Transp. Res. D 36, 107–120.
Watkins, C.J., Dayan, P., 1992. Q-learning. Mach. Learn. 8 (3–4), 279–292.
Xia, H., Boriboonsomsin, K., Barth, M., 2013. Dynamic eco-driving for signalized arterial corridors and its indirect network-wide energy/emissions benefits. J. Intell. Transp. Syst. 17 (1), 31–41.
Xu, S., Li, S.E., Zhang, X., Cheng, B., Peng, H., 2015. Fuel-optimal cruising strategy for road vehicles with step-gear mechanical transmission. IEEE Trans. Intell. Transp. Syst. 16 (6), 3496–3507.
Yan, F., Winijkul, E., Jung, S., Bond, T.C., Streets, D.G., 2011. Global emission projections of particulate matter (PM): I. Exhaust emissions from on-road vehicles. Atmos. Environ. 45 (28), 4830–4844.
Zarkadoula, M., Zoidis, G., Tritopoulou, E., 2007. Training urban bus drivers to promote smart driving: A note on a Greek eco-driving pilot program. Transp. Res. D 12 (6), 449–451.
Zhao, T., Hachiya, H., Niu, G., Sugiyama, M., 2011. Analysis and improvement of policy gradient estimation. In: Advances in Neural Information Processing Systems. pp. 262–270.