A Novel Framework Combining MPC and Deep Reinforcement Learning With Application to Freeway Traffic Control

Dingshan Sun, Anahita Jamshidnejad, and Bart De Schutter, *Fellow, IEEE*

***Abstract*— Model predictive control (MPC) and deep reinforcement learning (DRL) have been developed extensively as two independent techniques for traffic management. Although the features of MPC and DRL complement each other very well, few of the current studies consider combining these two methods for application in the field of freeway traffic control. This paper proposes a novel framework for integrating MPC and DRL methods for freeway traffic control that is different from existing MPC-(D)RL methods. Specifically, the proposed framework adopts a hierarchical structure, where a high-level efficient MPC component works at a low frequency to provide a baseline control input, while the DRL component works at a high frequency to modify online the output generated by MPC. The control framework, therefore, needs only limited online computational resources and is able to handle uncertainties and external disturbances after proper learning with enough training data. The proposed framework is implemented on a benchmark freeway network in order to coordinate ramp metering and variable speed limits, and the performance is compared with standard MPC and DRL approaches. The simulation results show that the proposed framework outperforms standalone MPC and DRL methods in terms of total time spent (TTS) and constraint satisfaction, despite model uncertainties and external disturbances.**

***Index Terms*— Freeway network management, model predictive control, deep reinforcement learning, hierarchical structure.**

I. Introduction

THE ever-growing number of vehicles worldwide is challenging current traffic systems. Especially during morning or evening rush hours, congestion easily occurs due to insufficient road capacity. Traffic jams do not only increase commute time for individuals, but also create negative impacts on society, including environmental, economic and health issues due to the large amount of emissions and the loss of productive time. Constructing new lanes and expanding the freeway network can alleviate these issues. However, this is not always feasible due to space, financial, or environmental restrictions. Efficient management of traffic on the existing infrastructure is a promising alternative to improve traffic efficiency and safety. Among freeway control measures, ramp metering (RM) [1] and variable speed limits (VSLs) [2] are the most widely used strategies, which have been shown to substantially decrease the travel delay in various real-world implementations [3], [4]. These two control measures can be either used independently or coordinated together within a control method, such as model predictive control (MPC) [5] or deep reinforcement learning (DRL) [6]. MPC and DRL are two powerful control techniques and have been studied extensively in the literature. These two methods have been applied to freeway traffic control successfully, but they also come along with their shortcomings.

MPC is a model-based optimal control approach, and its theory of stability and feasibility has become mature since 1990s [7]. It is widely applied in industry and many other fields, because of its robustness and ability to explicitly deal with input and state constraints, thus satisfying safety requirements, which is a crucial concern in many real-world applications. However, an accurate mathematical model is usually required for MPC to guarantee the closed-loop performance, while acquiring such a model is commonly not possible in practice. In particular, large-scale and complex systems, such as freeway networks, lead to highly nonlinear and non-convex optimization problems with multiple variables, which are difficult to solve in real time [8]. Even though some efficient MPC approaches have been developed to improve the computational efficiency of MPC [9], [10], [11], [12], the optimality and satisfaction of state constraints cannot always be guaranteed in case of model mismatches and external disturbances. Robust and stochastic MPC methods [13], including tube MPC [14] and scenario-based MPC [15], can address uncertainties to some extent. However, these methods require assumptions or descriptions of the uncertainties that are often difficult to validate.

DRL is a recent technique that has shown its success and potential in the field of control, including intelligent traffic control. Unlike conventional reinforcement learning algorithms, artificial neural networks are deployed in DRL to deal with large-dimension state and action spaces. This addresses the so-called issue of the curse of dimensionality [16].

Manuscript received 26 January 2023; revised 26 August 2023 and 9 December 2023; accepted 10 December 2023. Date of publication 2 January 2024; date of current version 2 July 2024. This work was supported in part by the European Research Council (ERC) under the European Union's Horizon 2020 Research and Innovation Program through ERC Advanced Grant CLariNet under Grant 101018826 and in part by the China Scholarship Council (CSC) under Grant 201806230254. The Associate Editor for this article was L. Qi. *(Corresponding author: Dingshan Sun.)*

Dingshan Sun and Bart De Schutter are with the Delft Center for Systems and Control, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: d.sun-1@tudelft.nl; b.deschutter@tudelft.nl).

Anahita Jamshidnejad is with the Department of Control and Operations, Faculty of Aerospace Engineering, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: a.jamshidnejad@tudelft.nl).

Digital Object Identifier 10.1109/TITS.2023.3342651

TABLE I
COMPARISON OF MPC AND DRL CHARACTERISTICS

	MPC	DRL
Need a model	Yes	No
Developed stability theory	Yes	No
Developed feasibility theory	Yes	No
Handling constraints explicitly	Yes	No
Adaptive to uncertainties	No	Yes
Online computational time	High	Low
Offline computation time	Low	High

Nevertheless, DRL still suffers from several challenges in real-world applications [17]. For example, safety constraints are of significant importance in operation of real-world systems, while satisfaction of constraints cannot be guaranteed during the learning phase and in implementation of DRL. In addition, the sample efficiency issue and delayed reward for large-scale systems (e.g., for traffic networks) remain considerable challenges for DRL that are still active research topics [17].

Both MPC and DRL have their advantages and disadvantages, and they complement each other well (see Table I). On the one hand, MPC suffers from degraded performance due to model uncertainties and external disturbances. Moreover, large-scale systems introduce multiple variables and long prediction horizons, which make MPC computationally intractable in real time. On the other hand, DRL can naturally cope with uncertainties, and tackle infinite prediction horizons with negligible online computational resources. However, it is usually time-consuming to train a well-performing DRL agent from scratch, especially for complex systems. Although there are clear potential benefits in combining MPC and DRL, very limited work has been done to explore the synergy between these two methods. In addition, very little work has been done to apply combined MPC-(D)RL algorithms in the field of traffic management. One of the representative studies is [18], which applied a model-reference framework that utilizes MPC and deep Q-network algorithm to urban traffic signal control.

The current paper contributes to the state-of-the-art by proposing a novel framework for combining MPC and DRL, and by applying it to traffic management of freeway networks. To be more specific:

1) Different from the previous work [18], the newly proposed MPC-DRL framework adopts a hierarchical structure in order to incorporate the advantages of both MPC and DRL approaches. The combined framework can learn from the environment, while providing basic control performance. By taking advantage of the dynamic model knowledge and environment information, the framework can deal with uncertainties and improve sample efficiency of the learning process. In particular, an efficient MPC controller operates at the upper control level with a low control frequency to provide initial optimality while explicitly incorporating the constraints. Meanwhile, a DRL agent works at the lower control level with a high control frequency in order to modify the MPC outputs, and to compensate for model mismatches that may affect MPC. Because of the hierarchical structure and multi-frequency control strategy, the proposed framework achieves a good balance between computational tractability and control performance.
2) The resulting MPC-DRL framework is implemented on a benchmark freeway network, and the results validate the effectiveness of the proposed method. In particular, the objective function of MPC and the reward function of DRL are designed properly, such that the two components are complementary with each other. In addition to MPC, DRL addresses the state and input constraints by introducing penalties on the constraint violation in the reward function. Simulation results show that the combined MPC-DRL outperforms other controllers in terms of control performance, constraint satisfaction, and computational efficiency.

The rest of this paper is organized as follows: Section II summarizes related work about MPC and DRL and their application in freeway traffic management, as well as the latest MPC-DRL algorithms and their applications. Section III presents and provides details on the novel MPC-DRL framework that is proposed in this paper. Section IV gives a case study that implements MPC, DRL, and the proposed MPC-DRL framework on the same benchmark network. Finally, Section V concludes the paper and proposes topics for future work.

II. RELATED WORK

A large number of studies about traffic management of freeway networks exist in the literature, and a recent comprehensive survey is given in [19]. Among all the traffic control approaches, MPC and DRL have drawn significant attention because of their appealing features. MPC and DRL have been developed and applied for both freeway and urban traffic networks. As the case study in Section IV involves a freeway traffic network, we will mainly focus on MPC and DRL for freeway traffic control in this section.[1] After that, current research gaps regarding combined MPC-DRL methods are analyzed.

A. MPC for Freeway Traffic Control

The idea of utilizing rolling horizon optimization in traffic signal control was first introduced by Gartner [20], after which the suggestion of adopting MPC in traffic signal control was formally made by De Schutter and De Moor in [21]. Since then extensive studies about MPC have been carried out in the field of traffic control, including railway [22], urban [23], and freeway traffic networks [8]. Particularly, an RM strategy and a VSLs strategy was adopted in MPC for freeway traffic control in, respectively, [24] and [25]. These two control methods were first coordinated within MPC in the work of Hegyi et al. [8].

As an online optimization-based control method, MPC struggles with computational complexity, especially when the scale of the freeway network is large. Therefore, a large amount of efforts have been devoted to alleviate this issue. One major direction is to reduce the complexity of the

[1]It is, however, important to note that the novel MPC-DRL framework proposed in this paper can be applied to both freeway and urban traffic networks.

dynamic model of the freeway network, and many efficient mathematical models have been developed to describe traffic flow dynamics, such as METANET [26], and cell transmission model (CTM) [27].

The other direction to improve the computational efficiency of MPC is to simplify the problem by linearizing it, or by adopting efficient optimization techniques. For example, Zegeye et al. [9] employed the parameterized MPC technique to reduce the number of the decision variables of the optimization problem. By introducing state-feedback control laws, the control inputs can be described as a function of the states and several function parameters. Thus, only the parameters need to be optimized to obtain the control inputs. Jeschke et al. further extended this approach by using a grammatical evolution method to generate the state-feedback laws automatically, and apply it to urban traffic control [12]. Ferrara et al. [28] incorporated an event-trigger mechanism in the MPC framework to reduce the frequency of solving the optimization problems. Besides, the finite-horizon optimization problem within the MPC scheme is formulated as a mixed-integer linear programming problem that can be solved efficiently, thanks to the revised linear model obtained from CTM.

Despite the success that efficient MPC algorithms have achieved, MPC still suffers from issues that are caused by uncertainties, since it relies heavily on the prediction model. Mismatches between the (macroscopic) prediction model and the real-world traffic system, as well as the presence of external disturbances are inevitable, which deteriorate the closed-loop performance of MPC.

To address these issues, a few studies have considered robust MPC for freeway traffic control. For example, Liu et al. [29] utilized a scenario-based approach [15] to describe the uncertainties as a set of scenarios with their corresponding probabilities, including global uncertainties (e.g., global weather conditions) and local uncertainties (e.g., local weather conditions, local traffic compositions, and local demands at the origins). Coordinated with distributed MPC (DMPC), the resulting scenario-based DMPC improves the control performance for a large-scale freeway network considering some uncertainties. Nevertheless, current robust MPC algorithms for freeway traffic control require assumptions and simplifications about the uncertainties and disturbances that are usually hard to satisfy in practice. Moreover, the extra computational burden introduced by robust MPC methods is another issue. Therefore, developing efficient and uncertainty-resistant MPC algorithms for traffic management remains a challenging and urgent task.

B. DRL for Freeway Traffic Control

Reinforcement learning (RL) [30] is a machine learning technique that usually follows a two-stage procedure. In the first stage, the RL agent learns how to take actions by interacting with the environment/system (or a model of it), in order to maximize the notion of cumulative reward. After that, the trained RL agent is then implemented for control. RL is attracting more and more interest from the system and control community, since it can naturally deal with uncertainties and automatically learn a long-term optimal policy through interacting with the environment. However, the drawbacks of conventional RL are still obvious, which is called the curse of dimensionality [16]. As a result, existing studies that use conventional RL for traffic management can only deal with a small traffic network [31], [32].

The emergence of DRL algorithms significantly broadens the applications of RL and unlocks great potentials for various fields. The introduced neural networks in DRL can handle more complex state and action spaces [31]. DRL has also been studied for intelligent traffic signal control [33], and a recent survey is offered in [34]. In addition, DRL has been successfully applied in other problems. For example, Zhang et al. [35] used DRL to solve a dynamic travelling salesman problem, and achieved substantial improvement within a very short computation time, compared with other baseline approaches.

Despite the great progress in DRL techniques, the limitations of DRL mentioned in Section I still apply. Moreover, current work mainly focuses on urban traffic signal control, while relevant research about DRL for freeway traffic control is still limited [36], [37], [38]. To the best of our knowledge, [39] is the only paper that coordinates VSLs and RM with a DRL algorithm, where both DDPG and TD3 [40] algorithms are implemented and their performance is compared. It is also shown that a centralized DRL agent can handle a large freeway network with multiple VSL-RM hybrid controllers. In addition, very few studies considers the state constraints. For example, the queue length of the on-ramps should be constrained, otherwise it interferes with the connected urban road network and safety issues may occur. Moreover, although a lot of research has studied how to improve the practicability of learning-based methods, such as by training with real-world data or by pre-training (i.e., before implementation), there is still a huge gap between real-world deployment and simulator-based applications.

DRL methods have the potential to deal with uncertain environments, but they also suffer from the requirement of a prolonged training process (i.e., low sample efficiency), as well as the lack of performance and safety guarantees. How to maintain the positive features of DRL, while circumventing the drawbacks remains an interesting and relevant research topic.

C. Current Research Gaps in MPC-(D)RL Methods

Considering the features of both MPC and DRL, the idea of merging these two methods to exploit their complementary advantages sounds promising. Although a few studies have investigated this topic, current methods have their corresponding drawbacks and no one has implemented MPC-DRL methods in the field of traffic management, especially for freeway traffic control. Therefore, in this subsection, the latest work relevant to combined MPC-(D)RL methods are analyzed, which further motivates this paper.

The paper [41] is the earliest one that utilizes a value function to approximate the infinite-horizon objective function of MPC, where a Markov Decision Process (MDP),

which is a discrete-time stochastic control process, is used as the prediction model. Moreover, the prediction horizon is reduced to look only one step ahead, while accounting for the long-term value of the performance criteria. The value function can be learned gradually on-line using RL techniques, and meanwhile MPC operates with a simplified optimization problem to provide data samples. This work opened up a research direction to combine MPC and RL algorithms, and inspired consequent research. The method was extended to more general dynamics in [42], where two different value function approximations are used and implemented for various control examples, including the inverted pendulum, the double pendulum, and the acrobot. However, the learning process still struggles with a low sample efficiency and unsafe exploration issues. Arroyo et al. [43] further extended the method given in [41] to a realistic scenario for building energy management, by encoding domain knowledge. Then the initial complex MPC optimization problem is reformulated as an optimization problem with a prediction horizon of one step. In [43] a simulation model is extracted from the simulator via system identification, and is used as the prediction model for MPC, as well as for pre-training of the DRL agent. The simulation results show that the proposed RL-MPC approach can meet the state constraints and provide satisfying performance. However, it is not demonstrated in [43] whether or not the RL-MPC approach outperforms MPC in uncertain environments.

The above MPC-DRL combined algorithms can be categorized as objective function truncating methods. This can reduce the on-line computational complexity of MPC, while RL is used to handle the uncertain environment. Nevertheless, these algorithms still suffer from several issues. First, one-step ahead MPC optimizes the control input only for the next time step, and therefore can only guarantee the short-term safety constraints. Second, although the value function can include the constraints by introducing a penalty on constraint violations in the reward function, in this way the constraints become soft constraints that do not necessarily provide guarantees. Third, an inaccurate system model is still used for the one-step ahead optimization of MPC, which influences the optimality of the performance. Fourth, optimizing the joint objective function and value function can be quite challenging, due to the nonlinearity and non-convexity introduced by the neural networks.

Another direction to connect MPC with RL is developed by Gros and Zanon. In [44] they proposed to use a parameterized MPC scheme instead of deep neural networks to approximate the value function and policy for the RL agent. It is shown that the MPC scheme can guarantee the optimality of the learned policy by adjusting the objective function of MPC, even with an inaccurate system model. Furthermore, they extended the algorithm by utilizing robust MPC techniques to address the safety issue of RL [45]. The method is implemented with a Q-learning algorithm and the results show that the constraints are well handled. In fact, Gros and Zanon [44], [45] are basically using RL tools to solve the MPC problem by using the connection between the parameterized MPC and RL. However, how to parameterize the cost function of MPC is not considered in a structured way.

A different trend is to directly combine the control inputs of MPC and RL. The paper [46] proposed a framework that contains independent MPC and DDPG agents, in which the overall output is a weighted sum of the control inputs generated by MPC and DRL. The idea is to use MPC to play a guiding role by applying its control action directly to the system to obtain more effective data samples for the training of the DDPG agent, thus improving the sample efficiency. However, the weight parameter needs to be tuned by trail and error for various tasks, and the synergies between MPC and DRL are not considered nor analyzed. The state and input constraints are not considered either.

There is not yet an extensive comparison study about the MPC-RL algorithms discussed above, so it is still an open question that which approach surpasses the other, and in which cases. However, each algorithm is designed to address a specific issue or a particular task. The current paper develops a novel framework that combines MPC and DRL in a flexible way, i.e., it allows the designers to freely choose the detailed MPC and DRL schemes. The framework is also designed using a hierarchical structure with multiple operation rates, such that MPC and DRL can coordinate well with each other, making the framework applicable for various complex applications. The proposed framework is tested for a freeway traffic control problem from [8], and the performance is compared with standard MPC, DRL methods, and advanced MPC methods.

III. Combined MPC-DRL Framework

This section presents the proposed MPC-DRL control framework. Section III-A gives an intuitive description of the framework from a high-level point-of-view. Section III-B defines the MPC and the DRL modules. Section III-C details the learning algorithm of the framework. The mathematical notations used in this section are presented and defined in Table II.

A. MPC-DRL Framework

As illustrated in Figure 1, the proposed MPC-DRL framework has a hierarchical structure. The MPC module operates at the high level to provide a basic control input that is optimized over the prediction window based on the objective function of MPC with the associated nominal model and the predicted traffic demands. The objective function is given according to the control purpose (e.g., minimizing TTS), and the state and input constraints are considered explicitly during the optimization. In practice, the MPC output $\boldsymbol{u}_{\mathrm{b}}$ is not optimal, mainly due to the mismatch between the prediction model and the real system, as well as due to the error in the predicted demands. Accordingly, the state constraints cannot be guaranteed. Note that MPC performs with a larger control sampling time T_{c} than the simulation sampling time T_{s}, such that the number of the optimization variables is substantially reduced, even with a long prediction horizon. Therefore the online optimization problem of MPC is computationally tractable.

In order to improve the optimality of the MPC output and to avoid severe constraint violations, the DRL module works at the lower level to modify the MPC output $\boldsymbol{u}_{\mathrm{b}}$ during the

TABLE II
MATHEMATICAL NOTATIONS USED FOR THE COMBINED MPC-DRL FRAMEWORK

Notation	Definition
k_{s}	Simulation sampling step counter of the system
k_{d}	Control step counter of the DRL module, which also corresponds to the control step of the controlled system
k_{c}	Control step counter of the MPC module
T_{s}	Simulation sampling time between two simulation sampling steps of the system
T_{d}	Control sampling time of the DRL module
T_{c}	Control sampling time of the MPC module
T_{ini}	Time to initialize the freeway network before control for the simulation
T	The duration of the total simulation time period
$\boldsymbol{u}_{\mathrm{b}}(k_{\mathrm{c}})$	Output of MPC module at control step k_{c}
$\boldsymbol{u}_{\mathrm{s}}(k_{\mathrm{s}})$	Output of MPC module at simulation sampling step k_{s}
$\boldsymbol{u}_{\mathrm{rl}}(k_{\mathrm{d}})$	Action of DRL generated by the actor network at control step k_{d}
$\boldsymbol{u}'_{\mathrm{rl}}(k_{\mathrm{d}})$	Action of DRL generated by the target actor network at control step k_{d}
$\boldsymbol{u}_{\mathrm{c}}(k_{\mathrm{d}})$	Final control input given to the system at control step k_{d}
$\boldsymbol{x}(k_{\mathrm{s}})$	Real traffic state at simulation sampling step k_{s}
$\hat{\boldsymbol{x}}(k_{\mathrm{s}})$	Predicted traffic state at simulation sampling step k_{s}
$\hat{\boldsymbol{d}}(k_{\mathrm{s}})$	Predicted traffic demands at simulation sampling step k_{s}
$\boldsymbol{d}(k_{\mathrm{s}})$	Real traffic demands at simulation sampling step k_{s}
$N_{\mathrm{p,c}}$	Prediction horizon length counted in terms of the MPC control step
$N_{\mathrm{p,s}}$	Prediction horizon length counted in terms of the simulation sampling step
F	Dynamic freeway model
$\boldsymbol{x}_{\mathrm{rl}}(k_{\mathrm{d}})$	State vector for the DRL agent at control step k_{d}
$r(\cdot,\cdot)$	Reward of the DRL agent for taking an action $\boldsymbol{u}_{\mathrm{rl}}(k_{\mathrm{d}})$ when at a state $\boldsymbol{x}_{\mathrm{rl}}(k_{\mathrm{d}})$
y_i	Learning target of the DRL agent for data sample i
R	Experience replay buffer of the DRL agent
N	Size of mini-batch sampled from the replay buffer
n	Number of steps to look ahead for the reward in the DRL algorithm
$w_n(k_{\mathrm{d}})$	Random noise added to the DRL actions at control step k_{d} for exploration
w_u	Scaling parameter of the DRL actions
w_p	Penalty weight in the DRL reward function

learning process by interacting with the real system. The state space of the DRL agent includes the freeway states $\boldsymbol{x}$ and the MPC output $\boldsymbol{u}_{\mathrm{b}}$ (see (7)), while the reward function is designed to complement the objective function of MPC, such that these two modules can collaborate to optimize the overall objective. In addition, the traffic demands are also fed into the DRL agent, and a penalty on the constraint violation is added to the reward function. The action $\boldsymbol{u}_{\mathrm{rl}}$ of DRL has the same dimension as $\boldsymbol{u}_{\mathrm{b}}$, but its elements have smaller magnitudes. The components of the DRL module are defined in detail in Section III-B. The update algorithms of the network parameters in the figure are presented in Section III-C.

Assume that the model of the freeway dynamic is discrete-time with a simulation sampling time T_{s}, and the DRL module works with a control sampling time T_{d}. Then the relationship between T_{s}, T_{d}, and T_{c} are described as:

$$T_{\mathrm{c}} = m_1 \cdot T_{\mathrm{d}} = m_1 \cdot m_2 \cdot T_{\mathrm{s}}, \quad m_1, m_2 \in \mathbb{N}^+, m_1 > 1. \tag{1}$$

Note that for the sake of simplicity and brevity in the notations, we assume that the simulation sampling step, DRL control step, and MPC control step coincide (see Figure 2). Therefore, the overall control input of the combined framework is a

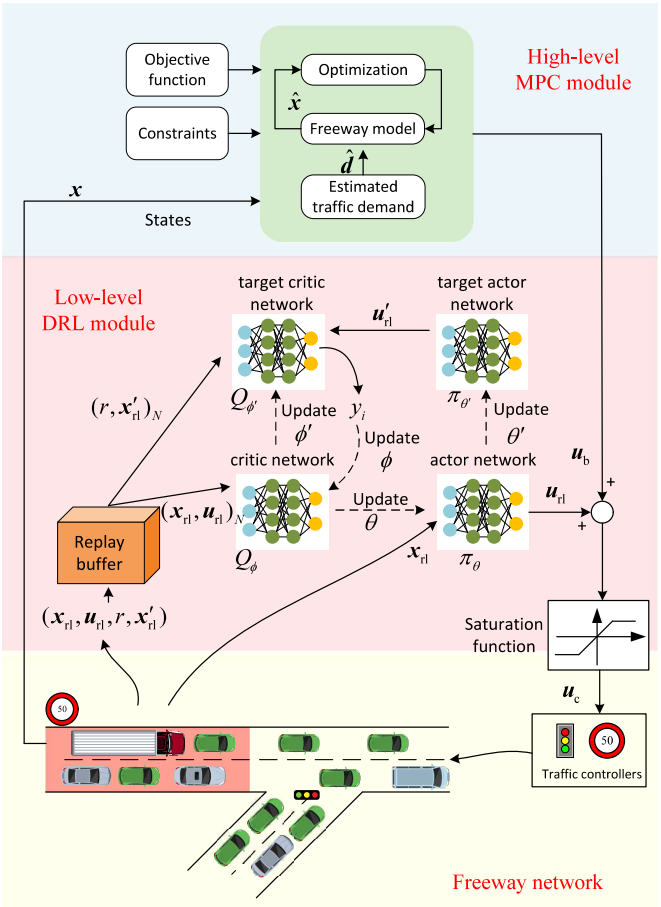

Fig. 1. Block diagram of the hierarchical MPC-DRL control framework, in which $\boldsymbol{x}'_{\mathrm{rl}}$ represents the state measured at the next control step.

combination of $\boldsymbol{u}_{\mathrm{b}}$ and $\boldsymbol{u}_{\mathrm{rl}}$, which is updated every T_{d} time units. For control step k_{d} that corresponds to the MPC control step k_{c} (i.e., $k_{\mathrm{d}}T_{\mathrm{d}} \in [k_{\mathrm{c}}T_{\mathrm{c}}, (k_{\mathrm{c}}+1)T_{\mathrm{c}})$), the overall control input is given by:

$$\boldsymbol{u}_{\mathrm{c}}(k_{\mathrm{d}}) = \mathrm{sat}(\boldsymbol{u}_{\mathrm{rl}}(k_{\mathrm{d}}) + \boldsymbol{u}_{\mathrm{b}}(k_{\mathrm{c}})), \tag{2}$$

where a saturation function $\mathrm{sat}(\cdot)$ is used to guarantee that the additive control input $\boldsymbol{u}_{\mathrm{c}}(k_{\mathrm{d}})$ satisfies the bound constraints, and is defined in element-wise by:

$$\mathrm{sat}(u) = \begin{cases} u_{\max}, & \text{if } u > u_{\max} \\ u_{\min}, & \text{if } u < u_{\min} \\ u, & \text{otherwise,} \end{cases} \tag{3}$$

with $u_{\min}$ and $u_{\max}$ the minimal and maximal allowed values for the corresponding elements in the control inputs for the freeway network, and $\boldsymbol{u}_{\mathrm{b}}(k_{\mathrm{c}})$ the corresponding MPC output. Figure 2 illustrates the different time scales of MPC and DRL control sampling time, and how $\boldsymbol{u}_{\mathrm{rl}}$ modifies $\boldsymbol{u}_{\mathrm{b}}$.

B. Detailed Description of the Framework

The details of the MPC and DRL modules are provided in this section.

1) MPC Module: A standard MPC procedure is performed within the MPC module, where a nominal model F is used to describe the freeway dynamic. The details of the freeway model are given in Appendix B. The system states $\boldsymbol{x}$ are

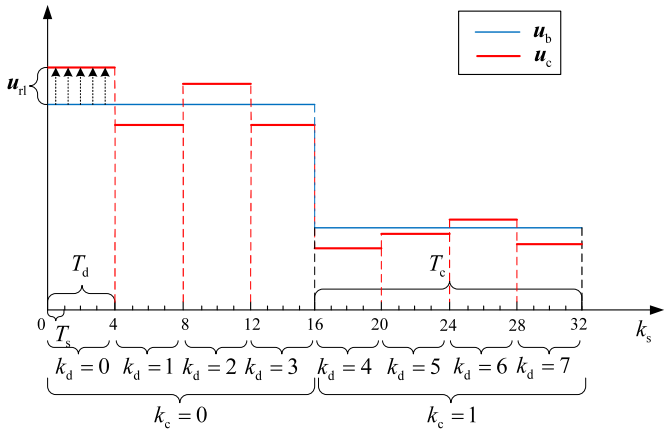

Fig. 2. Illustration of different time scales and how DRL modifies the MPC output.

updated every simulation sampling step k_s. The simulation sampling steps that correspond to the MPC control step k_c are given by:

$$\{k_c m, k_c m + 1, \ldots, k_c m + m - 1\}, \tag{4}$$

where $m = m_1 m_2$. Thus at simulation sampling steps $k_s = k_c m$, the real states of the freeway network are measured and are fed into the MPC module. The following optimization problem is solved at every control step k_c:

$$\min_{\tilde{\boldsymbol{u}}_b(k_c), \tilde{\boldsymbol{x}}(k_c)} \sum_{\ell=1}^{N_{p,s}} J(k_c m + \ell)$$
$$\text{s.t.} \quad \text{(A.1)-(A.4)}, \tag{5}$$

where $J(k_s)$ represents the predicted objective function value (e.g., TTS) during the time interval $[k_s T_s, (k_s + 1)T_s)$, $\tilde{\boldsymbol{u}}_b(k_c)$ denotes the variables to be optimized over the prediction window of length $N_{p,c}$, and $\tilde{\boldsymbol{x}}(k_c)$ denotes the predicted future states at control step k_c. In addition, $N_{p,s}$ and $N_{p,c}$ are the prediction horizon length counted in terms of the simulation sampling steps and MPC time steps, respectively, in which $N_{p,s} = N_{p,c} m$. For simplicity, the detailed formulation of the MPC module is present in Appendix A.

Due to the nonlinearity and non-smoothness of the traffic model, the resulting optimization problem is, in general, nonlinear and non-convex. Therefore, a nonlinear optimization solver, such as multi-start sequential quadratic programming (SQP), simulated annealing, or genetic algorithms [47] is required. After the above optimization problem is solved, the first element of the optimized control input $\tilde{\boldsymbol{u}}_b(k_c)$ is given to the DRL module.

2) DRL Module: Considering the freeway network as a Markov Decision Process (MDP), it can be represented by a five-tuple $\langle S, A, P, \mathcal{R}, \gamma \rangle$. The state space S, action space A, and reward distribution $\mathcal{R}$ are defined in this section. Furthermore, P denotes the transition probability among the states, and is implicitly defined by the freeway network model. Moreover, $\gamma \in [0, 1)$ is the user-defined discount factor on the future rewards. The DRL module operates at the low level with a higher frequency than the MPC module. The control sampling time T_d of the DRL module is larger than the simulation sampling time T_s, in order to avoid a too frequent change in the control inputs to the freeway network. Therefore, the simulation sampling steps that correspond to the DRL control step k_d includes:

$$\{k_d m_2, k_d m_2 + 1, \ldots, k_d m_2 + m_2 - 1\}, \tag{6}$$

where m_2 is defined in (1). The state, action, and reward of DRL are updated every control step k_d, and are defined as it follows.

State $\boldsymbol{x}_{rl}(k_d) \in S$: The state space of DRL should consist of all necessary information of the framework. Since deep neural networks are employed in DRL, the states fed into the input layer are normalized to the same order of magnitude in order to facilitate the learning. Thus, we have:

$$\boldsymbol{x}_{rl}(k_d) = [\bar{\boldsymbol{x}}^\top(k_d m_2), \bar{\boldsymbol{u}}_s^\top(k_d m_2), \bar{\boldsymbol{d}}^\top(k_d m_2), \bar{\boldsymbol{u}}_c^\top(k_d - 1)]^\top, \tag{7}$$

where $\bar{\boldsymbol{x}}(k_d m_2)$, $\bar{\boldsymbol{u}}_s(k_d m_2)$, $\bar{\boldsymbol{d}}(k_d m_2)$, and $\bar{\boldsymbol{u}}_c(k_d - 1)$ are, respectively, the normalized states of the freeway network, the MPC outputs, the real demands at simulation sampling step $k_d m_2$, and the overall control input of the framework at previous control step $k_d - 1$. Note that the fourth element is added to provide extra knowledge about the overall control input of the combined framework, which can be beneficial to avoid severe fluctuation in the control inputs.

Action $\boldsymbol{u}_{rl}(k_d) \in A$: The action $\boldsymbol{u}_{rl}$ is used to modify the MPC output $\boldsymbol{u}_b$. Therefore, they have the same dimension, i.e., $\dim \boldsymbol{u}_{rl} = \dim \boldsymbol{u}_b$. For simplicity, it is assumed that the action space is also continuous.

Note that action $\boldsymbol{u}_{rl}$ is generated from the output layer of the DNNs, and the original values of its elements are between $[-1, 1]$. Thus these values are scaled back to the real control inputs before they are added. Moreover, the elements of $\boldsymbol{u}_{rl}$ have a smaller magnitude than those of $\boldsymbol{u}_b$, such that $\boldsymbol{u}_b$ dominates the control input in this framework and provides basic performance, while $\boldsymbol{u}_{rl}$ is an ancillary control input that acts at a higher frequency and aims at improving the performance. Furthermore, $\boldsymbol{u}_{rl}$ meets the following inequalities that defines the action space A:

$$-w_u \Delta \boldsymbol{U} \le \boldsymbol{u}_{rl} \le w_u \Delta \boldsymbol{U}, \tag{8}$$

where $\Delta \boldsymbol{U} = \boldsymbol{u}_{max} - \boldsymbol{u}_{min}$, with $\boldsymbol{u}_{max}$ the upper bound and $\boldsymbol{u}_{min}$ the lower bound of MPC output, and $w_u \in [0, 1)$ is the scaling parameter that determines to what extent $\boldsymbol{u}_{rl}$ influences $\boldsymbol{u}_b$.

Reward $r(\boldsymbol{x}_{rl}(k_d), \boldsymbol{u}_{rl}(k_d)) \in \mathcal{R}$: In order to coordinate MPC and DRL to achieve the optimal performance, the reward function should include the objective function J of the MPC module:

$$r(\boldsymbol{x}_{rl}(k_d), \boldsymbol{u}_{rl}(k_d)) = \sum_{k=1}^{m_2} \big(- J(k_d m_2 + k) - w_p P_s(k_d m_2 + k) \big), \tag{9}$$

where P_s denotes the state constraint violation, and $w_p > 0$ is the penalty weight parameter. Let $r_t(k_d)$ be an equivalent representation of $r(\boldsymbol{x}_{rl}(k_d), \boldsymbol{u}_{rl}(k_d))$, which denotes the

observed reward based on the traffic condition during DRL control step k_d. In order to evaluate J, the relevant state $\boldsymbol{x}(k_\mathrm{s})$ can be measured at every simulation sampling step, and the MPC output $\boldsymbol{u}_\mathrm{s}(k_\mathrm{s})$ can be obtained from (A.4). Moreover, $P_s(k_\mathrm{s})$ can be calculated directly based on state $\boldsymbol{x}(k_\mathrm{s})$, for $k_\mathrm{s} = k_\mathrm{d}m_2 + 1, \ldots, k_\mathrm{d}m_2 + m_2$. The reward is a negative value, and thus $\mathcal{R}$ is the set of negative numbers.

The deep actor-critic algorithms are considered to train the framework, among which Deep Deterministic Policy Gradient (DDPG) [48] is chosen for the DRL agent, which is an off-policy and model-free algorithm that can deal with continuous state and action spaces, and has been implemented successfully in many freeway traffic studies (see, e.g., [38], [39], [49]).

Remark 1: The standard MPC procedure within the high-level MPC module can be replaced with any efficient MPC variants, such as parameterized MPC or DMPC for large-scale freeway networks. The DDPG agent can easily be extended to arbitrary off-policy DRL algorithms that can deal with continuous state and action spaces.

C. Algorithm for Training the Framework

The goal of learning is to train a policy π, such that the expected return at state $\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d})$ after taking action $\pi(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}))$ is maximized. The expected return for action $\pi(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}))$ taken at state $\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d})$ is given by:

$$\begin{aligned} &Q^{\pi}\left(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}), \pi(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}))\right) \\ &= \mathbb{E}_{r,\boldsymbol{x}_\mathrm{rl}\sim E}\left[\sum_{k=0}^{\infty}\gamma^k r(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}+k), \boldsymbol{u}_\mathrm{rl}(k_\mathrm{d}+k))\right] \\ &= \mathbb{E}_{r,\boldsymbol{x}_\mathrm{rl}\sim E}\big[r(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}), \boldsymbol{u}_\mathrm{rl}(k_\mathrm{d})) \\ &\quad + \gamma Q^{\pi}\left(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}+1), \pi(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}+1))\right)\big], \end{aligned} \tag{10}$$

where the subscript $r, \boldsymbol{x}_\mathrm{rl} \sim E$ implies that the transitions among the states in the environment are stochastic. Note that the return depends on the chosen actions, and thereafter on the policy π. In DDPG, both the return and the policy π are approximated by deep neural networks, which are notated as $Q_\phi^\pi(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}), \boldsymbol{u}_\mathrm{rl}(k_\mathrm{d}))$ and π_θ, respectively. They are also known as the critic and the actor, and ϕ and θ represent the parameters of the corresponding neural networks, as shown in Figure 1. In addition, the target network technique is used, which introduces two target critic and actor deep neural networks, corresponding to ϕ' and θ', that are updated in a slower pace in order to improve the stability of the training process. Experience replay is also utilized to remove correlations in the observation sequence, and to provide better learning convergence, and a replay buffer R is used to store the agent's experience. For more details, the readers are referred to [48].

Instead of the traditional one-step temporal-difference (TD) target in DDPG, we use the n-step TD method [30], i.e.:

$$y_{k_\mathrm{d}} = r_n(k_\mathrm{d}) + \gamma^n Q_{\phi'}^{\pi}(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}+n), \boldsymbol{u}'_\mathrm{rl}(k_\mathrm{d}+n)), \tag{11}$$

in which $\boldsymbol{u}'_\mathrm{rl}(k_\mathrm{d}+n) = \pi_{\theta'}(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}+n))$, and the n-step reward is given by:

$$r_n(k_\mathrm{d}) = \sum_{k=0}^{n-1}\gamma^k r(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}+k), \boldsymbol{u}_\mathrm{rl}(k_\mathrm{d}+k)), \tag{12}$$

where $\boldsymbol{u}_\mathrm{rl}(k_\mathrm{d}) = \pi_\theta(\boldsymbol{x}_\mathrm{rl}(k_\mathrm{d}))$. Then the loss function for the critic is given by:

$$L(\phi) = \frac{1}{N}\sum_i \left(y_i - Q_\phi^\pi(\boldsymbol{x}_\mathrm{rl}(i), \boldsymbol{u}_\mathrm{rl}(i))\right), \tag{13}$$

where i is the index of the data points of a mini-batch of size N that is sampled randomly from the experience replay buffer. The critic parameters ϕ are therefore updated by minimizing the loss function with the Adaptive Moment Estimation (Adam) optimizer [50].

The benefits of using n-step TD here are fourfold:

1) A freeway network is a large-scale system with time delays, which means the control measures only take effect after a period of time. Thus, looking n steps into the future can better evaluate the quality of the actions taken.
2) The optimization of the DDPG agent considers the reward for n future steps, which coincides with the predicted objective function in MPC. In practice, taking $n = N_\mathrm{p,s}/m_2$ makes the look-ahead time of DDPG and MPC the same, and thus these two modules cooperate better.
3) Looking n steps ahead makes the learning process more efficient than the one-step TD method of the conventional DDPG algorithm, where the update is only based on bootstrapping from the value of the state one step later [30].
4) By introducing future rewards in (11), there is no need to predict future demand information as the MPC module does. Therefore, the state space definition (7) is simpler and has a smaller dimension.

After the optimization of the critic network, the actor is subsequently updated by maximizing the return Q_ϕ^π based on the policy gradient. More details can be found in [48].

One advantage of DDPG as an off-policy algorithm is that its exploration policy is independent from the learning process, which means that a stochastic exploration is allowed. In this context, the Ornstein-Uhlenbeck model [51] is used to produce the noise $w_n(k_\mathrm{d})$ for exploration (i.e., added on the DRL actions), where the magnitude decays with time step k_d. The overall learning algorithm of the MPC-DRL framework is summarized in Algorithm 1. Note that the total simulation time is supposed to be T.

IV. Case Study

The proposed MPC-DRL framework is now implemented and evaluated via a benchmark freeway network from [8]. METANET is adopted to model this network, for which the readers are referred to [8] and [26]. Model uncertainties and external disturbances are introduced into the model to represent the real-world system, as illustrated in Section IV-A. Furthermore, the proposed MPC-DRL framework is compared with standalone MPC and DRL methods, and one advanced MPC method (i.e., parameterized MPC [9]). In this case study, the performance criteria consist of TTS of all the vehicles for the entire traffic network, total waiting time (TWT) of all the queues, minimum traffic speed during the total simulation

Algorithm 1 Hierarchical MPC-DRL Framework Algorithm for Freeway Traffic Control

Initialize critic and actor networks Q_ϕ^π and π_θ with parameters ϕ and θ
Initialize target network $Q_{\phi'}$ and $\pi_{\theta'}$ with parameters ϕ' and θ'
Initialize experience replay buffer R
for episode from 1 to M **do**
  Initialize the empty traffic network with initial traffic demands for T_{ini} time units
  for $k_\text{c} = 0$ to $\frac{T}{T_\text{c}} - 1$ (MPC outer loop) **do**
    Observe current traffic state $\boldsymbol{x}(k_\text{c}m)$ from the environment, and estimate the traffic demands $\hat{\boldsymbol{d}}(k_\text{c}m + k), k = 0, 1, \ldots, N_{\text{p,s}}$
    Perform high-level MPC with freeway model F and prediction horizon $N_{\text{p,s}}$, by solving optimization problem (5)-(A.3)
    Pass the optimized MPC output $\boldsymbol{u}_\text{b}(k_\text{c})$ to the low-level DRL module
    for $k_\text{d} = k_\text{c}m_1$ to $(k_\text{c}+1)m_1 - 1$ (DRL inner loop) **do**
      Receive state $\boldsymbol{x}_\text{rl}(k_\text{d})$ according to (7)
      Select action $\boldsymbol{u}_\text{rl}(k_\text{d}) = \pi_\theta(\boldsymbol{x}_\text{rl}(k_\text{d})) + w_n(k_\text{d}))$
      Combine the output of MPC and RL with a saturation function using (2)
      for $k_\text{s} = k_\text{d}m_2, \ldots, k_\text{d}m_2 + m_2 - 1$ **do**
        Execute action $\boldsymbol{u}_\text{c}(k_\text{d})$ in the freeway network, with the real traffic demand $\boldsymbol{d}(k_\text{s})$
      end for
      Observe reward $r_\text{t}(k_\text{d})$ and new state $\boldsymbol{x}_\text{rl}(k_\text{d}+1)$
      Store transition $(\boldsymbol{x}_\text{rl}(k_\text{d}), \boldsymbol{u}_\text{rl}(k_\text{d}), r_\text{t}(k_\text{d}), \boldsymbol{x}_\text{rl}(k_\text{d}+1))$ in R (the transitions are stored in order)
      Sample a mini-batch of N transitions from R randomly, each of which contains n steps: $\boldsymbol{x}_\text{rl}(i), \boldsymbol{u}_\text{rl}(i), r_\text{t}(i), \boldsymbol{x}_\text{rl}(i+1), \ldots, \boldsymbol{x}_\text{rl}(i+n-1), \boldsymbol{u}_\text{rl}(i+n-1), r_\text{t}(i+n-1), \boldsymbol{x}_\text{rl}(i+n)$
      Update the critic network Q_ϕ^π by minimizing the loss function $L(\phi)$ with (11)-(13)
      Update the actor network π_θ by the sampled policy gradient [48]
      Update the target networks: $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$, $\phi' \leftarrow \tau\phi + (1-\tau)\phi'$
    end for
  end for
end for

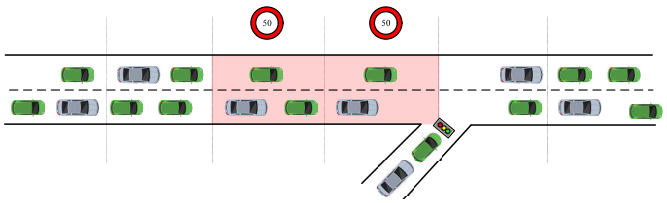

Fig. 3. The benchmark freeway network with one metered on-ramp and two segments with speed limits (marked in red) used for the case study.

time, constraint violations of the queues on the lanes, and the online computation time. All the simulations were conducted in Matlab version 2022a running on a PC with an Intel Xeon Quad-Core E5-1620 V3 CPU with a clock speed of 3.5 GHz.

A. Setup

1) Freeway Traffic Network: A benchmark network is taken from [8]. Note that this benchmark network has also been used in other freeway traffic studies [52], [53], [54]. As shown in Figure 3, the network consists of two origins

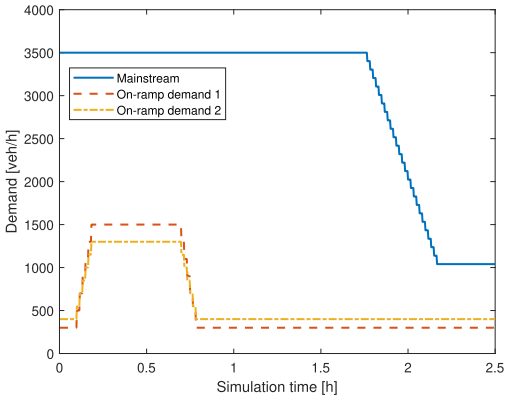

Fig. 4. Predicted traffic demands used in the case study.

(i.e., one mainstream and one on-ramp) and one destination. The length of the main stretch is 6 km, which is divided into 6 segments of 1 km. The mainstream has two lanes with a capacity of 2000 veh/h each, and its maximal allowed queue length is 200 veh. The on-ramp has one lane with the capacity of 2000 veh/h, and the maximum on-ramp queue length is 100 veh. The network parameters are taken from [8] and the same mathematical notations are used here. The real parameters are assumed unknown in this case study, and the estimated values for these parameters are used in the prediction model. Both the real and the estimated parameter values are given in Table III.

2) Demand Scenario: Two typical demand scenarios as shown in Figure 4 similar to [8] are considered in order to evaluate the controllers. These two demand scenarios have the same profile for the mainstream. Without control both of them can cause severe traffic congestion, and they are suitable to examine the control effectiveness of both ramp metering and variable speed limits in this freeway network. The freeway network is initially empty, and is next simulated with a constant demand at 3000 veh/h for the mainstream and 500 veh/h for the on-ramp for a period of 10 min, before the control simulations start.

3) Noises: To reproduce the stochastic phenomena of the traffic network, random noise with a Gaussian distribution is added to the demands of both mainstream and on-ramp. To fully evaluate the ability of the controllers to resist uncertainties, we consider three noise levels, i.e., low-level noise, medium-level noise, and high-level noise. More specifically, the noise levels have the following distributions:

- Low-level noise: $\mathcal{N}(0, 75)$ for mainstream demand, and $\mathcal{N}(0, 30)$ for on-ramp demand;
- Medium-level noise: $\mathcal{N}(0, 150)$ for mainstream demand, and $\mathcal{N}(0, 60)$ for on-ramp demand;
- High-level noise: $\mathcal{N}(0, 225)$ for mainstream demand, and $\mathcal{N}(0, 90)$ for on-ramp demand.

B. Controllers

In this case study, the following controllers are implemented and compared: standalone MPC, standalone DRL (with n-step TD), high-frequency standalone MPC, parameterized MPC, and combined MPC-DRL framework (with n-step TD). The

TABLE III
REAL AND ESTIMATED VALUES FOR FREEWAY NETWORK PARAMETERS IN THE CASE STUDY

	T[s]	τ[s]	κ[veh/km/lane]	η[km^2/h]	a_m	σ	v_{free}[km/h]	ρ_{crit}[veh/km/lane]	α	$\rho_{\max}$[veh/km/lane]	L_m[m]
Real	10	18	40	60	1.867	0.0122	102	33.5	0.1	180	1000
Estimates	10	14.5	48	50	2.160	0.01	102	37.5	0.08	150	800

simulation parameters are $T_{\text{s}} = 10$ s, $T_{\text{d}} = 60$ s, $T_{\text{c}} = 300$ s, $T_{\text{ini}} = 600$ s, $T = 9000$ s. These parameters apply for all the controllers. Similar to [8] and [55], the ramp metering rate ranges from 0 to 1, and the variable speed limits range from 20 km/h to v_{free}, which is 102 km/h. Therefore, $u_{\min} = [20\ 20\ 0]^\top$ and $u_{\max} = [102\ 102\ 1]^\top$. Both control actions are continuous.

1) Standalone MPC Controller: The objective function (5) used in the MPC controller is written as

$$J(k_{\text{s}}) = w_{\text{TTS}} J_{\text{TTS}}(k_{\text{s}}) + w_{\mathcal{D}} \left\| \boldsymbol{u}_{\text{s}}(k_{\text{s}}) - \boldsymbol{u}_{\text{s}}(k_{\text{s}} - 1) \right\|_2^2,$$

where $J_{\text{TTS}}(k_{\text{s}})$ denotes the TTS value predicted for total simulation time period $[k_{\text{s}}T_{\text{s}}, (k_{\text{s}} + 1)T_{\text{s}})$, the second term imposes a penalty on the fluctuations between consecutive control inputs, and the weights are selected as $w_{\text{TTS}} = 1$, $w_{\mathcal{D}} = 0.4$. The prediction window is 10 min, which means $N_{\text{p,c}} = 2$, $N_{\text{p,s}} = 60$. The prediction model used by the MPC controller is the METANET model with the estimated parameter values in Table III, and the predicted demands used by MPC are shown in Figure 4. The control inputs include one ramp metering rate and two variable speed limits, which are constrained by the lower and upper bounds given before. In this controller, the optimization problem is solved every 300 s, which is at a low frequency.

For simplicity, the control problem is transcribed into an optimization problem by single-shooting [56]. The resulting optimization problem is nonlinear and non-convex, so the Matlab `fmincon` function with the SQP algorithm are used to solve the optimization problem. In order to avoid getting stuck in local optima, multiple starting points are used to solve the optimization problem (i.e., 30 for this case study[2]). In order to achieve a trade-off between the computational accuracy and efficiency, the `fmincon` stopping criteria for the cost function tolerance, step tolerance, and constraint tolerance are selected to be 10^{-2}.

2) Standalone DRL (With n-Step TD): The standalone DRL agent in this case study shares the same definition as the DRL module in Section III-B.2. Therefore, the dimensions of the state space and action space are 30 and 3, respectively. The actor network contains one input layer of size 30, one output layer of size 3, and two inner layers with 256 neurons for both layers. Accordingly, the critic network has two input layers, in which one layer that corresponds to the states is of size of 30 and is followed by a layer with 256 neurons, and the other input layer that corresponds to the actions is of size 3 and is followed by a layer with 128 neurons. Both input layers are connected to two consecutive inner layers with 256 and 128 neurons, respectively. The size of the output layer, which generates the Q-values of the state-action pairs, is 1. ReLU activation functions are used in all the neural networks. Moreover, the reward function consists of the objective function defined for the MPC controller and a penalty for constraint violation with weight $w_p = 10$. The actions of the standalone DRL are the same as the MPC controller. The other DRL parameters that are tuned during the learning process are given in Table IV. These parameters apply for both conventional standalone DRL and n-step TD DRL.

[2]This number is obtained via several experiments, such that it achieves a good trade-off between the computational efficiency and optimality.

TABLE IV
PARAMETERS USED FOR DRL AGENT TRAINING

Parameter	Value
Maximal episodes M	3000
Mini-batch size N	512
Experience replay buffer R size	$2 \cdot 10^5$
Discounter factor γ	0.99
Learning rate (both actor and critic networks)	0.001
Target network update rate τ	0.01
Noise w_n standard deviation	0.3
Noise w_n decay rate	$5 \cdot 10^{-6}$

3) High-Frequency Standalone MPC Controller: This is a high-frequency version of the standalone MPC controller, in which the main differences are that the control sampling time is $T_{\text{c}} = 60$ s and the optimization problem is thus solved at every 60 s. This means that this controller requires higher online computational efforts, but also results in a better control performance.

4) Parameterized MPC Controller: The parameterized MPC (PMPC) controller developed for integrated VSL-RM freeway traffic control [9] is used in this case study. PMPC is an efficient MPC controller: since the number of optimization variables are reduced because of the introduced parameterized control law, PMPC can reduce the online computation time significantly with comparable control performance, compared with a conventional MPC controller. For more details, the reader can refer to [9] and the references therein. In this case study, the PMPC controller shares settings with the high-frequency standalone MPC controller, including the objective function, prediction window, control sampling time, optimization parameters, etc.

5) Combined MPC-DRL Framework (With n-Step TD): The combined MPC-DRL framework (with n-step TD) consists of an MPC module that is the same as the standalone MPC controller, and a DRL module that is similar to the standalone DRL (with n-step TD). The action space of the MPC-DRL framework is different from the standalone DRL, because of the scaling parameter $w_u = 0.4$, which means that according to (8) the action bounds of the DRL module within the MPC-DRL framework are $\pm$ 0.32 for variable speed limits and $\pm$ 0.4 for ramp metering rate. The training parameters in Table IV also apply for the combined MPC-DRL framework,

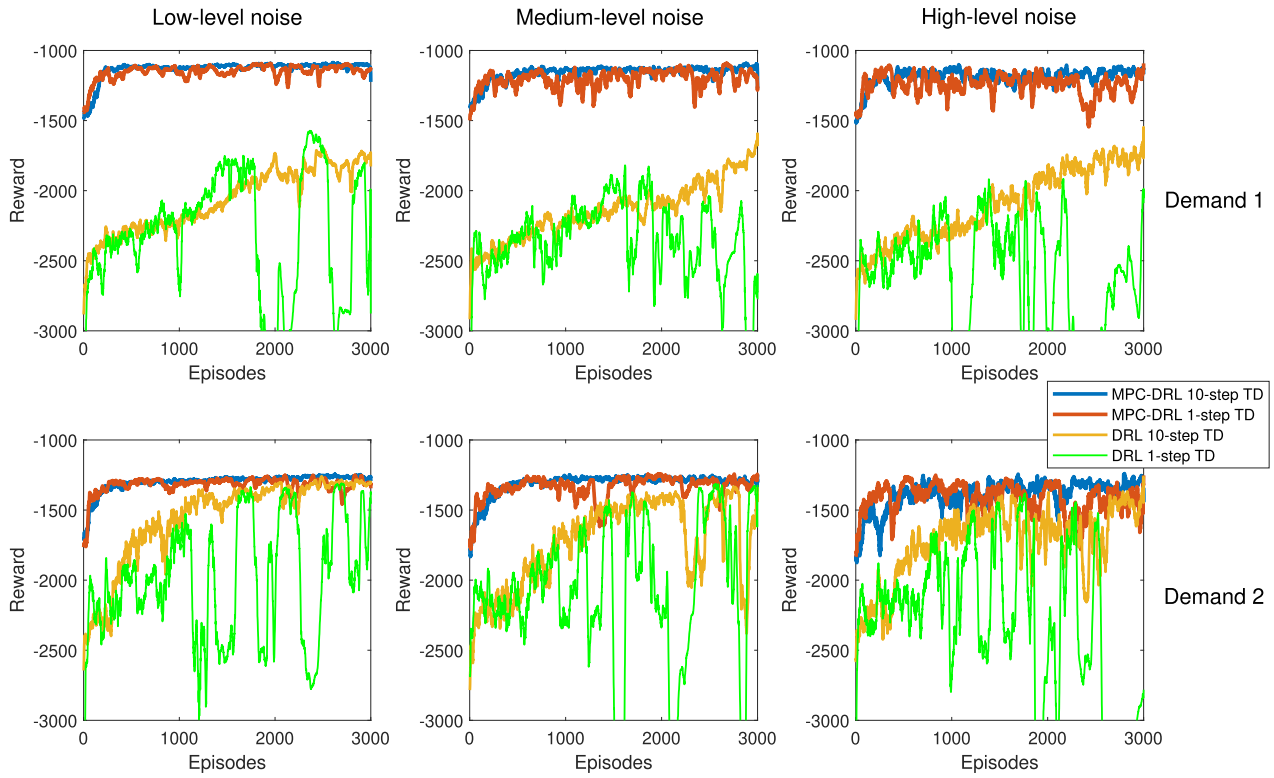

Fig. 5. Learning performance of standalone DRL and combined MPC-DRL with and without 10-step TD for different scenarios.

except for the noise w_n, which has a smaller standard deviation of 0.2 and a decay rate of 2×10^{-5}. Furthermore, $n = 10$ is chosen for the DRL module, which makes the look-ahead time of the DRL module the same as the prediction horizon of the MPC controller.

C. Results for the Learning Process

The standalone DRL (with 10-step TD) and the combined MPC-DRL framework (with 10-step TD) are both trained independently over the stochastic environment (i.e., the benchmark freeway network with stochastic demands), with 3000 episodes for each run. Each episode contains a simulation interval of 9000 s with the mentioned stochastic demands. In the plots, the episode rewards have first been smoothed by a moving average filter of size 21 to better present the learning progress. The learning performance is presented in Figure 5. There are 6 scenarios in total, which are:

- Scenario 1: Low-level noise & demand 1;
- Scenario 2: Medium-level noise & demand 1;
- Scenario 3: High-level noise & demand 1;
- Scenario 4: Low-level noise & demand 2;
- Scenario 5: Medium-level noise & demand 2;
- Scenario 6: High-level noise & demand 2.

Figure 5 shows that the learning curves of the combined MPC-DRL framework methods start with higher rewards and converge faster than the standalone DRL methods for all the scenarios. This indicates that the proposed framework has a better sample efficiency than the conventional DRL methods. The reason is that the MPC module within the MPC-DRL framework generates basic control inputs that provide baseline control performance and guide the DRL to learn, and the DRL module within the framework only requires a limited exploration space with smaller action bounds and thus requires less sample data, compared with the standalone DRL agent.

Furthermore, the methods with 10-step TD have a better learning performance than the ones without 10-step TD, which validates the advantages of the n-step TD method. In particular, the framework with 10-step TD has a more stable learning curve than the one without 10-step TD, and this phenomenon is more obvious as the noise level increases. This implies that a DRL module with a similar prediction window to the MPC module can cooperate better within the combined MPC-DRL framework. In addition, DRL without 10-step TD fails to converge for all the scenarios, while DRL with 10-step TD learns better and converges for scenario 4, 5, and 6. For scenario 1, 2, and 3, the DRL with 10-step TD fails to converge within 3000 episodes due to the low sample efficiency.

D. Results for the Implementations

According to the learning performance, only the trained standalone DRL controller with 10-step TD and the combined MPC-DRL controller with 10-step TD are implemented on the benchmark freeway network, and their control performance is evaluated in terms of TTS which represents the global traffic efficiency, constraint violations, and online CPU time. In addition, the total waiting time (TWT) of all the vehicles in a queue is considered for comparison, which indicates the congestion degree of the traffic network. The minimum

TABLE V

COMPARISON OF THE CONTROL PERFORMANCE FOR DIFFERENT CONTROLLERS FOR DIFFERENT SCENARIOS, IN TERMS OF TTS, TWT, CONSTRAINT VIOLATION, MEAN COMPUTATION TIME, AND MAX COMPUTATION TIME, IN WHICH '−' MEANS THAT THE CORRESPONDING ITEM IS NOT APPLICABLE TO THE CONTROLLER, AND HF MPC REFERS TO HIGH-FREQUENCY MPC

Scenarios	Performance	No control	Standalone MPC	Standalone DRL	HF MPC	PMPC	MPC-DRL framework
Scenario 1	Total time spent [veh·h]	1456.23	1407.47	1466.31	1368.45	1404.34	1375.87
	Total waiting time [veh·h]	353.58	344.17	365.15	308.24	426.40	321.46
	Minimum speed [km/h]	12.65	13.77	12.16	14.09	11.69	13.11
	Constraint violation [%]	90.63	38.79	58.86	18.64	174.98	9.97
	Mean computation time [s]	-	0.82	-	21.23	2.32	0.70
	Max computation time [s]	-	1.72	-	52.74	6.48	1.43
Scenario 2	Total time spent [veh·h]	1463.38	1400.39	1397.08	1374.92	1370.96	1388.29
	Total waiting time [veh·h]	371.07	356.99	318.37	320.74	425.69	332.78
	Minimum speed [km/h]	12.66	12.88	13.20	13.99	12.03	13.04
	Constraint violation [%]	91.77	34.82	46.18	21.11	165.39	12.62
	Mean computation time [s]	-	0.79	-	21.96	2.39	0.78
	Max computation time [s]	-	1.84	-	51.80	6.75	1.59
Scenario 3	Total time spent [veh·h]	1475.94	1441.28	1428.17	1368.76	1339.53	1376.76
	Total waiting time [veh·h]	378.69	344.65	354.70	315.90	425.78	324.21
	Minimum speed [km/h]	12.53	13.97	11.64	13.56	12.18	13.14
	Constraint violation [%]	100.32	44.82	15.23	22.30	224.76	14.25
	Mean computation time [s]	-	0.89	-	22.50	2.86	0.75
	Max computation time [s]	-	2.43	-	52.87	7.68	1.73
Scenario 4	Total time spent [veh·h]	1351.66	1303.80	1316.97	1231.47	1239.91	1265.21
	Total waiting time [veh·h]	239.57	235.57	267.69	223.39	279.06	236.30
	Minimum speed [km/h]	15.21	16.70	15.16	17.18	15.65	17.16
	Constraint violation [%]	27.57	3.68	13.19	0.92	123.15	0.00
	Mean computation time [s]	-	0.74	-	22.85	2.56	0.68
	Max computation time [s]	-	1.53	-	54.61	6.05	1.00
Scenario 5	Total time spent [veh·h]	1360.15	1308.30	1361.59	1222.81	1239.15	1255.17
	Total waiting time [veh·h]	250.77	238.09	259.65	225.29	276.58	225.46
	Minimum speed [km/h]	15.20	15.87	15.90	17.21	14.47	16.99
	Constraint violation [%]	31.42	9.66	3.59	1.88	135.54	0.27
	Mean computation time [s]	-	0.81	-	23.08	2.94	0.69
	Max computation time [s]	-	1.87	-	55.39	7.20	1.12
Scenario 6	Total time spent [veh·h]	1330.32	1296.31	1333.68	1252.80	1207.26	1266.48
	Total waiting time [veh·h]	235.84	233.94	243.32	232.45	265.39	224.39
	Minimum speed [km/h]	14.82	16.07	15.22	16.99	14.84	16.19
	Constraint violation [%]	26.96	10.48	0.95	2.55	99.90	0.00
	Mean computation time [s]	-	0.82	-	23.01	2.64	0.70
	Max computation time [s]	-	1.92	-	56.08	6.96	1.05

traffic speed on the links during the total simulation time is compared, which represents the worst congestion degree. The standalone MPC controller, high-frequency MPC, PMPC, and the no-control case (i.e., no ramp metering or speed limit) are included for comparison. Because of the stochastic feature of the network, the experiments for each controller are repeated 10 times independently with random demands in order to evaluate the control performance. The quantitative simulation results are present in Table V, in which the constraint violation is the ratio of maximal exceeded queue length with respect to the maximum allowed queue length, the mean computation time is the average time required for the optimization process per control step (every 300 s), and the max computation time corresponds to the maximum computation time per step over all the control steps (every 300 s).

As shown in Table V, all the controllers can improve the traffic control performance with regard to the no-control case in terms of TTS, expect for the standalone DRL controller (Scenario 1, 5, and 6). This is due to the insufficient learning process and the low sample efficiency of the standalone DRL agent. The standalone MPC controller can improve the control performance compared to the no-control case with limited online computational complexity, in terms of TTS, TWT, minimum speed, and constraint satisfaction. The high-frequency MPC controller can further improve the performance of the standalone MPC controller, which, however, comes at the cost of a substantially higher online computational complexity (more than 20 times higher). The PMPC controller can reduce the online computational complexity of the high-frequency MPC controller significantly, and further reduce the TTS for several scenarios (Scenario 2, 3, and 6). However, the PMPC controller performs worse for constraint satisfaction, TWT, and minimum speed.

The proposed MPC-DRL framework outperforms both the standalone MPC controller and the standalone DRL controller, in terms of TTS, TWT, and constraint satisfaction, with similar online computational complexity for all the considered scenarios. The minimum speed of the proposed framework is slightly lower than the standalone MPC or standalone DRL controller in some scenarios. This is because more vehicles in the queues on both mainstream and on-ramp are allowed to enter the network in order to avoid constraint violation, thus reducing

the flow speed. In general, the results show that the framework has learned from interacting with the environment, and that the DRL module within the framework can compensate for the model uncertainties and external disturbances and in this way, provides extra optimality.

Although the high-frequency MPC controller can achieve the best TTS, TWT, and minimum speed performance for most scenarios, it still suffers from the uncertainties and noise in the demand, which is reflected through the constraint violation for all the scenarios. For the scenarios with traffic demand 2 (i.e., Scenario 4, 5, and 6), the constraint violation can be avoided (see the MPC-DRL framework controller). However, the high-frequency MPC controller still has constraint violations, while the MPC-DRL framework controller can guarantee constraint satisfaction. For the scenarios with traffic demand 1 (i.e., Scenario 1, 2, and 3), the traffic congestion is more severe, and constraint violation is inevitable. In this case, the MPC-DRL framework controller can further reduce the constraint violation compared to the high-frequency MPC controller, at the price of slightly higher TTS and TWT and lower minimum speed with significantly less online computational burden. This indicates that, in addition to the smaller action space and high sample efficiency of the combined MPC-DRL framework, the penalty on the constraint violations within the reward function of the DRL module, which coincides with the state constraints within the optimization problem of the MPC module, also contributes to avoiding or relieving the constraint violations. So in the proposed MPC-DRL framework, the MPC and the DRL modules complement each other during both the learning process and the implementation stage, which results in a better sample efficiency and control performance in terms of TTS, TWT, minimum speed and constraint violation, with very limited online computational complexity.

V. Conclusion and Topics for Future Work

This paper has developed a novel framework combining MPC and DRL for freeway traffic control. Since MPC and DRL each suffer from their own shortcomings and their characteristics complement each other well, it is beneficial to merge these two methods. The proposed MPC-DRL framework inherits the ability of DRL in learning from the environment to deal with uncertainties, and the ability of MPC in using the model information to provide basic performance. Specifically, the novel framework has a hierarchical structure, in which an MPC controller works at a high level with a lower frequency, while the DRL agent operates at a low level with a high frequency. An additional advantage of the proposed framework is that it requires less computational efforts compared to conventional MPC, thanks to the lower control frequency of the MPC module.

A simulation study has been conducted on a benchmark freeway network with model uncertainties and stochastic traffic demands. The proposed MPC-DRL framework (with n-step TD), standalone MPC, high-frequency MPC, PMPC and DRL (with n-step TD) were trained and implemented for this traffic network. The results of the case study showed that the proposed MPC-DRL framework outperforms MPC and DRL in terms of both the learning process and the control performance, and the n-step TD method can improve the learning-based controllers for large-scale traffic networks. Moreover, the combined framework is easy to implement and can also potentially be applied to other systems that struggle with model uncertainties and high computational complexity, such as for the control of urban traffic networks or for the energy management of smart buildings. Future research will be conducted on extending the current framework to control large-scale traffic networks by combining distributed MPC and multi-agent DRL [57].

Appendix

A. MPC Module Formulation

The mathematical details of the MPC module within the combined MPC-DRL framework are given as below:

$$\min_{\tilde{\boldsymbol{u}}_{\mathrm{b}}(k_{\mathrm{c}}),\tilde{\boldsymbol{x}}(k_{\mathrm{c}})} \sum_{\ell=1}^{N_{\mathrm{p,s}}} J(k_{\mathrm{c}}m+\ell)$$

$$\text{s.t.} \quad \hat{\boldsymbol{x}}(k_{\mathrm{c}}m+\ell+1) = F(\hat{\boldsymbol{x}}(k_{\mathrm{c}}m+\ell), \boldsymbol{u}_{\mathrm{s}}(k_{\mathrm{c}}m+\ell), \hat{\boldsymbol{d}}(k_{\mathrm{c}}m+\ell)), \quad \text{for } \ell = 0,\ldots,N_{\mathrm{p,s}}-1, \tag{A.1}$$

$$\hat{\boldsymbol{x}}(k_{\mathrm{c}}m+\ell) \in \mathcal{X}, \quad \text{for } \ell = 1,\ldots,N_{\mathrm{p,s}}, \tag{A.2}$$

$$\boldsymbol{u}_{\mathrm{b}}(k_{\mathrm{c}}+k) \in \mathcal{U}, \quad \text{for } k = 0,1,\ldots,N_{\mathrm{p,c}}-1, \tag{A.3}$$

$$\boldsymbol{u}_{\mathrm{s}}((k_{\mathrm{c}}+k)m+\ell)) = \boldsymbol{u}_{\mathrm{b}}(k_{\mathrm{c}}+k), \quad \text{for } \ell = 0,1,\ldots,m-1, k = 0,1,\ldots,N_{\mathrm{p,c}}-1, \tag{A.4}$$

where $\tilde{\boldsymbol{u}}_{\mathrm{b}}(k_{\mathrm{c}}) = [\boldsymbol{u}_{\mathrm{b}}^{\top}(k_{\mathrm{c}}),\ldots,\boldsymbol{u}_{\mathrm{b}}^{\top}(k_{\mathrm{c}}+N_{\mathrm{p,c}}-1)]^{\top}$ denotes the variables to be optimized over the prediction window of length $N_{\mathrm{p,c}}$, with $\boldsymbol{u}_{\mathrm{b}}(k_{\mathrm{c}})$ the MPC outputs (e.g. ramp metering rates or variable speed limits) at control step k_{c}. Moreover, $\boldsymbol{u}_{\mathrm{s}}(k_{\mathrm{s}})$ is the MPC output at simulation sampling step k_{s}, and $\tilde{\boldsymbol{x}}(k_{\mathrm{c}}) = [\hat{\boldsymbol{x}}^{\top}(k_{\mathrm{c}}m+1),\ldots,\hat{\boldsymbol{x}}^{\top}(k_{\mathrm{c}}m+N_{\mathrm{p,s}})]^{\top}$ with $\hat{\boldsymbol{x}}(k_{\mathrm{s}})$ denoting the predicted future state at simulation sampling step k_{s}. Besides, $\hat{\boldsymbol{d}}(k_{\mathrm{s}})$ contains the estimated traffic demands at simulation sampling step k_{s}. Equation (A.1) represents the evolution of the system states driven by the freeway dynamics F. Equations (A.2) and (A.3) represent the constraints on the MPC state set $\mathcal{X}$ and the MPC output set $\mathcal{U}$, respectively. Since the frequency of system sampling is higher than that of MPC output generating, (A.4) maps the MPC outputs to every simulation sampling step during the prediction window, such that the output can be implemented in system dynamic (A.1). The freeway model F is introduced in detail in Appendix B.

B. Freeway Model: METANET

METANET [26], [58] is a macroscopic freeway traffic model that achieves a good trade-off between efficiency and accuracy and has been widely used [8], [53], [55], [59]. The model used in this paper is taken from [8]. In the METANET model, a freeway link is divided into several segments, each of which is described by segment traffic density $\rho_{m,i}(k)$, mean speed $v_{m,i}(k)$, and traffic volume or outflow $q_{m,i}(k)$, where m is the link index, i is the segment index, and k is the simulation

sampling step. The fundamental relationship between speed, flow, and density is given by:

$$q_{m,i}(k) = \rho_{m,i}(k) v_{m,i}(k) \lambda_m, \tag{A.5}$$

where λ_m denotes the number of lanes in the freeway link. The density equation is:

$$\rho_{m,i}(k+1) = \rho_{m,i}(k) + \frac{T}{L_m \lambda_m}(q_{m,i-1}(k) - q_{m,i}(k)), \tag{A.6}$$

where T is the simulation sampling interval (e.g., 10 s), and L_m is the length of the segment. The speed update equation is:

$$\begin{aligned} v_{m,i}(k+1) = v_{m,i}(k) &+ \frac{T}{\tau}\left(V(\rho_{m,i}(k)) - v_{m,i}(k)\right) \\ &+ \frac{T}{L_m} v_{m,i}(k)\left(v_{m,i-1}(k) - v_{m,i}(k)\right) \\ &- \frac{\eta T}{\tau L_m}\frac{\rho_{m,i+1}(k) - \rho_{m,i}(k)}{\rho_{m,i}(k) + \kappa}, \end{aligned} \tag{A.7}$$

where τ, η, κ are model parameters and with

$$V\left(\rho_{m,i}(k)\right) = v_{\text{free},m} \exp\left[-\frac{1}{a_m}\left(\frac{\rho_{m,i}(k)}{\rho_{\text{crit},m}}\right)^{a_m}\right], \tag{A.8}$$

where $v_{\text{free},m}, a_m, \rho_{\text{crit},m}$ are model parameters. Origins (e.g., on-ramps and source links) are modeled with a simple queue model, and the queue length w_o equation is given by:

$$w_o(k+1) = w_o(k) + T\left(d_o(k) - q_o(k)\right), \tag{A.9}$$

where $d_o(k)$ is the traffic demand at step k, and the flow $q_o(k)$ is given by:

$$\begin{aligned} q_o(k) = \min\Bigg[& d_o(k) + \frac{w_o(k)}{T}, C_o r_o(k), \\ & C_o(k)\left(\frac{\rho_{\max,\mu} - \rho_{\mu,1}(k)}{\rho_{\max,\mu} - \rho_{\text{crit},\mu}}\right)\Bigg], \end{aligned} \tag{A.10}$$

where C_o is the on-ramp capacity under free-flow conditions, $r_o(k)$ is the ramp metering rate at step k, μ is the index of the link that the on-ramp is connected to, and $\rho_{\max,\mu}$ is the maximum density of link μ. For more details, the reader can refer to [8] and the references therein.

ACKNOWLEDGMENT

The authors would like to thank Dr. Qingrui Zhang of Sun Yat-sen University and Yun Li of the Delft University of Technology for the useful discussions and their valuable suggestions on the DRL implementation.

REFERENCES

[1] M. Papageorgiou and A. Kotsialos, "Freeway ramp metering: An overview," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 4, pp. 271–281, Dec. 2002.

[2] C. Lee, B. Hellinga, and F. Saccomanno, "Evaluation of variable speed limits to improve traffic safety," *Transp. Res. C, Emerg. Technol.*, vol. 14, no. 3, pp. 213–228, Jun. 2006.

[3] S. P. Hoogendoorn, J. Van Kooten, and R. Adams, "Lessons learned from field operational test of integrated network management in Amsterdam," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2554, no. 1, pp. 111–119, Jan. 2016.

[4] R. Horowitz et al., "Design, field implementation and evaluation of adaptive ramp metering algorithms," California PATH, Richmond, CA, USA, Res. Rep. UCB-ITS-PRR-2005-2, 2005.

[5] E. F. Camacho and C. B. Alba, *Model Predictive Control*. Cham, Switzerland: Springer, 2013.

[6] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.

[7] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, Jun. 2000.

[8] A. Hegyi, B. De Schutter, and H. Hellendoorn, "Model predictive control for optimal coordination of ramp metering and variable speed limits," *Transp. Res. C, Emerg. Technol.*, vol. 13, no. 3, pp. 185–209, Jun. 2005.

[9] S. K. Zegeye, B. De Schutter, J. Hellendoorn, E. A. Breunesse, and A. Hegyi, "A predictive traffic controller for sustainable mobility using parameterized control policies," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1420–1429, Sep. 2012.

[10] Y. Li, K. Hua, and Y. Cao, "Using stochastic programming to train neural network approximation of nonlinear MPC laws," *Automatica*, vol. 146, Dec. 2022, Art. no. 110665.

[11] J. Fu, A. Núñez, and B. D. Schutter, "A short-term preventive maintenance scheduling method for distribution networks with distributed generators and batteries," *IEEE Trans. Power Syst.*, vol. 36, no. 3, pp. 2516–2531, May 2021.

[12] J. Jeschke, D. Sun, A. Jamshidnejad, and B. De Schutter, "Grammatical-evolution-based parameterized model predictive control for urban traffic networks," *Control Eng. Pract.*, vol. 132, Mar. 2023, Art. no. 105431.

[13] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Syst. Mag.*, vol. 36, no. 6, pp. 30–44, Dec. 2016.

[14] W. Langson, I. Chryssochoos, S. V. Raković, and D. Q. Mayne, "Robust model predictive control using tubes," *Automatica*, vol. 40, no. 1, pp. 125–133, Jan. 2004.

[15] G. C. Calafiore and M. C. Campi, "The scenario approach to robust control design," *IEEE Trans. Autom. Control*, vol. 51, no. 5, pp. 742–753, May 2006.

[16] V. Mnih et al., "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.

[17] G. Dulac-Arnold et al., "Challenges of real-world reinforcement learning: Definitions, benchmarks and analysis," *Mach. Learn.*, vol. 110, no. 9, pp. 2419–2468, Sep. 2021.

[18] W. Remmerswaal, D. Sun, A. Jamshidnejad, and B. De Schutter, "Combined MPC and reinforcement learning for traffic signal control in urban traffic networks," in *Proc. 26th Int. Conf. Syst. Theory, Control Comput. (ICSTCC)*, Oct. 2022, pp. 432–439.

[19] S. Siri, C. Pasquale, S. Sacone, and A. Ferrara, "Freeway traffic control: A survey," *Automatica*, vol. 130, Aug. 2021, Art. no. 109655.

[20] N. H. Gartner, "OPAC: A demand-responsive strategy for traffic signal control," *Transp. Res. Rec.*, vol. 906, pp. 75–84, Jan. 1983.

[21] B. De Schutter and B. De Moor, "Optimal traffic light control for a single intersection," *Eur. J. Control*, vol. 4, no. 3, pp. 260–276, Jan. 1998.

[22] X. Liu, A. Dabiri, Y. Wang, and B. De Schutter, "Modeling and efficient passenger-oriented control for urban rail transit networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 3, pp. 3325–3338, Mar. 2023.

[23] A. Jamshidnejad, D. Sun, A. Ferrara, and B. De Schutter, "A novel bi-level temporally-distributed MPC approach: An application to green urban mobility," *Transp. Res. C, Emerg. Technol.*, vol. 156, Nov. 2023, Art. no. 104334.

[24] T. Bellemans, B. De Schutter, and B. De Moor, "Model predictive control for ramp metering of motorway traffic: A case study," *Control Eng. Pract.*, vol. 14, no. 7, pp. 757–767, Jul. 2006.

[25] A. Hegyi, B. DeSchutter, and J. Hellendoorn, "Optimal coordination of variable speed limits to suppress shock waves," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 1, pp. 102–112, Mar. 2005.

[26] A. Kotsialos, M. Papageorgiou, M. Mangeas, and H. Haj-Salem, "Coordinated and integrated control of motorway networks via non-linear optimal control," *Transp. Res. C, Emerg. Technol.*, vol. 10, no. 1, pp. 65–84, Feb. 2002.

[27] C. F. Daganzo, "The cell transmission model, Part II: Network traffic," *Transp. Res. B, Methodol.*, vol. 29, no. 2, pp. 79–93, Apr. 1995.

[28] A. Ferrara, S. Sacone, and S. Siri, "Event-triggered model predictive schemes for freeway traffic control," *Transp. Res. C, Emerg. Technol.*, vol. 58, pp. 554–567, Sep. 2015.

[29] S. Liu, A. Sadowska, and B. De Schutter, "A scenario-based distributed model predictive control approach for freeway networks," *Transp. Res. C, Emerg. Technol.*, vol. 136, Mar. 2022, Art. no. 103261.
[30] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[31] Z. Li, P. Liu, C. Xu, H. Duan, and W. Wang, "Reinforcement learning-based variable speed limit control strategy to reduce traffic congestion at freeway recurrent bottlenecks," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 3204–3217, Nov. 2017.
[32] M. Davarynejad, A. Hegyi, J. Vrancken, and J. van den Berg, "Motorway ramp-metering control with queuing consideration using Q-learning," in *Proc. 14th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2011, pp. 1652–1658.
[33] D. Sun, A. Jamshidnejad, and B. De Schutter, "Adaptive parameterized control for coordinated traffic management using reinforcement learning," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 5463–5468, 2023.
[34] A. Haydari and Y. Yilmaz, "Deep reinforcement learning for intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 11–32, Jan. 2022.
[35] Z. Zhang, H. Liu, M. Zhou, and J. Wang, "Solving dynamic traveling salesman problems with deep reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 4, pp. 2119–2132, Apr. 2023.
[36] M. Greguric, K. Kušic, F. Vrbanic, and E. Ivanjko, "Variable speed limit control based on deep reinforcement learning: A possible implementation," in *Proc. Int. Symp. ELMAR*, Sep. 2020, pp. 67–72.
[37] Y. Han, M. Wang, L. Li, C. Roncoli, J. Gao, and P. Liu, "A physics-informed reinforcement learning-based strategy for local and coordinated ramp metering," *Transp. Res. C, Emerg. Technol.*, vol. 137, Apr. 2022, Art. no. 103584.
[38] Y. Wu, H. Tan, L. Qin, and B. Ran, "Differential variable speed limits control for freeway recurrent bottlenecks via deep actor-critic algorithm," *Transp. Res. C, Emerg. Technol.*, vol. 117, Aug. 2020, Art. no. 102649.
[39] C. Wang, Y. Xu, J. Zhang, and B. Ran, "Integrated traffic control for freeway recurrent bottleneck based on deep reinforcement learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 15522–15535, Sep. 2022.
[40] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.
[41] R. R. Negenborn, B. De Schutter, M. A. Wiering, and H. Hellendoorn, "Learning-based model predictive control for Markov decision processes," *IFAC Proc. Volumes*, vol. 38, no. 1, pp. 354–359, 2005.
[42] M. Zhong, M. Johnson, Y. Tassa, T. Erez, and E. Todorov, "Value function approximation and model predictive control," in *Proc. IEEE Symp. Adapt. Dyn. Program. Reinforcement Learn. (ADPRL)*, Apr. 2013, pp. 100–107.
[43] J. Arroyo, C. Manna, F. Spiessens, and L. Helsen, "Reinforced model predictive control (RL-MPC) for building energy management," *Appl. Energy*, vol. 309, Mar. 2022, Art. no. 118346.
[44] S. Gros and M. Zanon, "Data-driven economic NMPC using reinforcement learning," *IEEE Trans. Autom. Control*, vol. 65, no. 2, pp. 636–648, Feb. 2020.
[45] M. Zanon and S. Gros, "Safe reinforcement learning using robust MPC," *IEEE Trans. Autom. Control*, vol. 66, no. 8, pp. 3638–3652, Aug. 2021.
[46] H. Xie, X. Xu, Y. Li, W. Hong, and J. Shi, "Model predictive control guided reinforcement learning control scheme," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.
[47] C. A. Floudas and P. M. Pardalos, *State of the Art in Global Optimization: Computational Methods and Applications*. Berlin, Germany: Springer, 2013.
[48] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
[49] M. Gregurić, K. Kušić, and E. Ivanjko, "Impact of deep reinforcement learning on variable speed limit strategies in connected vehicles environments," *Eng. Appl. Artif. Intell.*, vol. 112, Jun. 2022, Art. no. 104850.
[50] D. P. Kingma and J. Ba, "ADAM: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[51] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the Brownian motion," *Phys. Rev.*, vol. 36, p. 823, Sep. 1930.
[52] T. Schmidt-Dumont and J. H. van Vuuren, "Decentralised reinforcement learning for ramp metering and variable speed limits on highways," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 8, pp. 1–10, Aug. 2015.
[53] J. R. D. Frejo, A. Núñez, B. De Schutter, and E. F. Camacho, "Hybrid model predictive control for freeway traffic using discrete speed limit signals," *Transp. Res. C, Emerg. Technol.*, vol. 46, pp. 309–325, Sep. 2014.
[54] P. Chanfreut, J. M. Maestre, and E. F. Camacho, "Coalitional model predictive control on freeways traffic networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 11, pp. 6772–6783, Nov. 2021.
[55] J. R. D. Frejo and E. F. Camacho, "Global versus local MPC algorithms in freeway traffic control with ramp metering and variable speed limits," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1556–1565, Dec. 2012.
[56] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," in *Fast Motions in Biomechanics and Robotics*. Cham, Switzerland: Springer, 2006, pp. 65–93.
[57] M. Lv and N. Wang, "Distributed control for uncertain multi-agent systems with the powers of positive-odd numbers: A low-complexity design approach," *IEEE Trans. Autom. Control*, early access, 2023, doi: 10.1109/TAC.2023.3266986.
[58] A. Kotsialos, M. Papageorgiou, and A. Messner, "Integrated optimal control of motorway traffic networks," in *Proc. Amer. Control Conf.*, vol. 3, Mar. 1999, pp. 2183–2187.
[59] A. Hegyi and S. P. Hoogendoorn, "Dynamic speed limit control to resolve shock waves on freeways–field test results of the SPECIALIST algorithm," in *Proc. 13th Int. IEEE Conf. Intell. Transp. Syst.*, Sep. 2010, pp. 519–524.

Dingshan Sun received the master's degree from Shanghai Jiao Tong University, China, in 2018, and the Ph.D. degree from the Delft Center for Systems and Control, Delft University of Technology (TU Delft), Delft, The Netherlands, in October 2023. He is currently a Post-Doctoral Researcher with the Department of Transport and Planning, TU Delft. His research interests include model predictive control, reinforcement learning, and traffic network control.

Anahita Jamshidnejad received the Ph.D. degree (cum laude) from the Delft University of Technology (TU Delft), The Netherlands, in 2017. She is currently an Assistant Professor with the Faculty of Aerospace Engineering, TU Delft. Her main research interests include optimization theory in engineering problems, integrated control methods, and real-time model predictive control, with applications in autonomous robotic systems and road traffic.

Bart De Schutter (Fellow, IEEE) is currently a Full Professor with the Delft Center for Systems and Control, Delft University of Technology, The Netherlands. His current research interests include intelligent transportation and infrastructure systems, hybrid systems, and multi-level multi-agent control. He is an Associate Editor of IEEE TRANSACTIONS ON AUTOMATIC CONTROL and a Senior Editor of IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS.