Université Gustave Eiffel

Reinforcement Learning and Optimal Control Project

Course: Reinforcement Learning and Optimal Control
Master 2 Systèmes Intelligents et Applications (SIA)
Academic year: 2024-2025

D-DOC: DDPG-based Reinforcement Learning for Traffic Light Control under Delayed Observations and Continuous-action space

Members

Cong-Son DUONG 289223
cong-son.duong@edu.univ-eiffel.fr

Mohammed Adel DJELLOUL ABBOU 284198
djelloulabbou@edu.univ-eiffel.fr

Francky RASATA xxx
xxx

Nikethan xxx
xxx

Luca Uckermann xxx
xxx

Project Supervisor

Prof. Nadir Farhi
Researcher(HDR) , Univ Gustave Eiffel

Paris, 25 December 2024

Table of Contents

Digital Object Identifier xxx

D-DOC: DDPG-based Reinforcement Learning for Highway Ramp-Metering Control under Delayed Observations and Continuous-action space

CONG-SON DUONG[1], MOHAMMED ADEL DJELLOUL ABBOU[2], FRANCKY RASATAHARISOA [3], NIKETHAN NIMALAKUMARAN[4], LUCAS UCKERMANN[5], (Member, IEEE)

[1]National Institute of Standards and Technology, Boulder, CO 80305 USA (e-mail: author@boulder.nist.gov)
[2]Department of Physics, Colorado State University, Fort Collins, CO 80523 USA (e-mail: author@lamar.colostate.edu)
[3]Electrical Engineering Department, University of Colorado, Boulder, CO 80309 USA

Corresponding author: First A. Author (e-mail: author@ boulder.nist.gov).

This paragraph of the first footnote will contain support information, including sponsor and financial support acknowledgment. For example, "This work was supported in part by the U.S. Department of Commerce under Grant BS123456."

ABSTRACT Ramp metering is an effective strategy to reduce traffic congestion by controlling the rate of vehicles entering the highway freeway. However, practical ramp-metering systems often face challenges such as delayed sensor data and the need for continuous control signals. To address these challenges, we propose **D-DOC**, a DDPG-based Reinforcement Learning framework specifically designed for ramp-metering control under both delayed state information and continuous action spaces. Leveraging Deep Deterministic Policy Gradient, our approach adaptively learns optimal metering rates that minimize congestion on the ramp and maximize throughput on the highway freeway.

We had implemented the proposed method in the SUMO simulation environment and evaluate performance across two contrasting scenarios: (1) a constant scenario with static flows, and (2) a dynamic scenario exhibiting peak hours from 8:15 to 8:45. Key traffic metrics—including Traffic Flow, Average Speed, Density, and Travel Time—are gathered; to assess the quality of control. Comparisons are made against a no-control approach (fixed or pre-timed signals) as well as RL methods with decision-making strategies (e.g., ϵ-Greedy, DQN). Our results show that D-DOC effectively manages ramp inflows despite observation delays, maintaining higher flow rates, reducing congestion, and improving overall travel conditions. This study thus demonstrates the practicality and benefits of a DDPG-driven policy for real-time ramp metering under uncertain and continuous-control requirements.

The source code implementation of the model is available at: https://github.com/CongSon01/RL_Ramp

INDEX TERMS Reinforcement Learning, Intelligent Traffic Light Control, Deep Deterministic Policy Gradient (DDPG), Urban Traffic Management

I. INTRODUCTION

Traffic jams are still a major problem in cities, causing longer travel times, wasted fuel, and more pollution. One key factor is how vehicles merge onto highways from on-ramps. This merging can disrupt the flow of traffic on the highway and make congestion worse. Studies [1] have shown that vehicles entering from on-ramps can cause more variation in speeds on the highway, leading to frequent lane changes and drivers speeding up or slowing down, which worsens traffic flow. Generally, Ramp metering manages the flow of vehicles onto highways from on-ramps. Traditional ramp metering methods (e.g., fixed-time controls and pre-timed signals) typically use data from road sensors. While these methods have demonstrated effectiveness in reducing congestion and improving safety, they are limited by their inability to adapt to real-time traffic dynamics and their reliance on discrete control actions.

Recent advancements have explored the integration of

reinforcement learning (RL) techniques into ramp metering control to enhance adaptability and performance. In [2], an RL method is proposed to combine with Model Predictive Control (MPC) to manage on-ramp traffic, demonstrating improved congestion reduction and constraint satisfaction compared to traditional controllers. Additionally, In [3], deep RL algorithm is used to improve the performance in ramp metering applications, highlighting their potential in handling uncertainties and variable demands.

Despite these advancements, practical ramp metering systems often face challenges such as delayed observations due to sensor data latency and the necessity for continuous actuator signals to manage traffic flow effectively. To address these challenges, we propose D-DOC, a Deep Deterministic Policy Gradient (DDPG)-based reinforcement learning framework specifically designed for ramp metering control under conditions of delayed state information and continuous action spaces. Our approach introduces novel definitions of state, reward, and action functions to enhance system performance.

Through comprehensive experiments conducted in the SUMO simulation environment, our proposed model demonstrates significant improvements in traffic metrics, including increased traffic flow, higher average speeds, reduced density, and decreased travel time, even in scenarios with observation delays.

The main contributions of this paper are:

- **Modeling the Ramp Metering Problem:** We formulate the ramp metering control problem considering delayed observations and continuous action spaces, providing a realistic representation of practical traffic systems.
- **Integration of Reinforcement Learning Algorithms:** We incorporate the DDPG algorithm into the ramp metering problem and propose novel state, reward, and action functions to improve system performance.
- **Comprehensive Experimental Evaluation:** We conduct extensive evaluations across various scenarios, including normal traffic conditions, congestion, and delayed observations, to validate the effectiveness of our proposed approach.

The remainder of this paper is organized as follows: Section 2 reviews related work on reinforcement learning applications in ramp metering control. Section 3 provides preliminaries on highway ramp metering control and reinforcement learning approaches. Section 4 presents our proposed solution, detailing the state, action, and reward functions, and the integration of DDPG into the problem. Section 5 offers experimental evaluations based on three primary scenarios: normal conditions, congestion, and delayed observations. Finally, Section 6 concludes the paper and discusses potential directions for future research.

II. RELATED WORKS

Recent studies show that applying reinforcement learning (RL) to ramp metering has significantly improved traffic management. The chronological progression of these approaches reflects the increasing sophistication of models and their ability to address complex traffic scenarios.

Fares and Gomaa (2014) [4] introduced a reinforcement learning-based approach for freeway ramp metering, using traffic density as the primary state variable. Their discrete action model aimed to optimize traffic flow by maintaining freeway density below critical levels to prevent congestion. Simulation results demonstrated a 6.5% reduction in total travel time and a 6.74% increase in average speed compared to the base case without control measures. However, the applicability of their method was limited to specific traffic scenarios, indicating a need for broader validation across diverse conditions.

Yang et al. (2018) [5] proposed a deep reinforcement learning-based control framework targeting freeway weaving sections. By considering traffic flow parameters, their continuous action model improved overall traffic operations. The approach resulted in a notable improvement in traffic flow and a reduction in congestion at freeway bottlenecks. However, the model required extensive training data to achieve optimal performance, highlighting the need for substantial data collection efforts and the potential challenges in generalizing the model to different traffic environments.

Liu et al. (2020) [3] proposed the application of deep reinforcement learning (DRL) for ramp metering by utilizing visual traffic data as state variables, marking a significant departure from traditional loop detector inputs. Their model achieved a 22% reduction in average travel time and a 17% decrease in queue length compared to conventional fixed-time metering strategies. Despite these improvements, the model required high computational resources, making real-time deployment challenging, particularly in large-scale networks with variable traffic conditions.

Zhao et al. (2023) [6] introduced a novel DRL-based algorithm that incorporated traffic flow and density as key state inputs, focusing on discrete action spaces. Their results indicated a 19% enhancement in traffic throughput during peak hours and a 15% reduction in overall congestion levels relative to state-of-the-art ALINEA controllers. However, the model exhibited slower responsiveness to sudden traffic surges, which constrained its effectiveness in highly dynamic environments. This limitation highlights the ongoing need for models capable of faster adaptation to real-time traffic fluctuations.

Liu et al. (2023) [7] extended their earlier work by analyzing the robustness of DRL algorithms in ramp metering under adversarial conditions. By leveraging loop detector data, their model demonstrated resilience, outperforming traditional Q-learning-based approaches by 24% in terms of travel time optimization. Notably, the model reduced the frequency of stop-and-go conditions by 21%, contributing to smoother traffic flow. Nevertheless, the model's vulnerability to data manipulation posed a significant limitation, indicating the necessity for more robust defense mechanisms against potential cyber-attacks or sensor failures.

Paper(Year)	Paper Title	State Variables	Reward Metrics	Action	Results	Limitations
[4] (2014)	Freeway Ramp-Metering Control Based on Reinforcement Learning	Number of vehicles and ramp traffic signal	Freeway density around the critical density	Discrete	6.5% reduction in total travel time; 6.74% increase in average speed	Limited to specific traffic scenarios
[5] (2018)	A Deep Reinforcement Learning-Based Ramp Metering Control Framework	Traffic flow parameters	Traffic operation improvement	Continuous	Improvement in traffic flow; reduction in congestion	Requires extensive training data
[3] (2020)	A Deep Reinforcement Learning Approach for Ramp Metering Based on Traffic Video Data	Position matrix	Speed and the Queue length	Continuous	Lower travel times, shorter queues	High computational requirements
[6] (2023)	A Novel Ramp Metering Algorithm based on Deep Reinforcement Learning	Traffic flow, density	Traffic efficiency	Discrete	Enhanced traffic efficiency	Slow response to sudden changes
[7] (2023)	Analyzing Robustness of the Deep Reinforcement Learning Algorithm in Ramp Metering Applications	Loop detector data	Traffic flow optimization	Discrete	Outperforms traditional methods	Vulnerable to data attacks
[8] (2023)	Demonstration-Guided Deep Reinforcement Learning for Coordinated Ramp Metering and Perimeter Control	Traffic dynamics	Congestion alleviation	Continuous	Effective in large-scale networks	Dependence on demonstration quality
[2] (2023)	Reinforcement Learning with Model Predictive Control for Highway Ramp Metering	Traffic conditions, control variability, queue lengths	Congestion reduction, constraint satisfaction	Continuous	Improved performance over traditional controllers	Requires precise model tuning

TABLE 1: Comparison of Recent RL Approaches for Ramp Metering Control

Hu et al. (2023) [8] adopted a demonstration-guided approach to DRL, introducing coordinated ramp metering and perimeter control for large-scale networks. This model integrated real-time traffic dynamics and achieved a 26% improvement in congestion alleviation compared to fixed-time control methods. The coordinated strategy effectively minimized bottlenecks by synchronizing ramp meters and perimeter signals, resulting in a 20% decrease in vehicle idle times across highly congested freeway segments. However, the model's performance heavily depended on high-quality demonstration data, which introduced biases and limited generalizability to unseen traffic conditions.

Airaldi et al. (2023) [2] proposed a hybrid framework combining RL with Model Predictive Control (MPC) to address variability in traffic conditions and optimize ramp queue management. Their continuous action space model yielded a 23% increase in highway throughput and reduced ramp queue lengths by 18% compared to traditional ramp metering algorithms. This approach demonstrated the effectiveness of integrating predictive modeling into RL frameworks, but the results were contingent on precise model calibration. The necessity for extensive parameter tuning introduced practical deployment challenges, especially in networks with fluctuating demand patterns.

Table 1 provides a detailed overview of these contributions, summarizing the state variables, reward metrics, action types, and key results. Collectively, these studies reflect the growing potential of reinforcement learning to revolutionize traffic control at freeway ramps by enhancing throughput, mitigating congestion, and reducing travel times. However, the inherent limitations—such as high computational costs, sensitivity to data quality, and slow adaptation to abrupt changes—underscore the need for ongoing research to refine these methodologies further.

III. PRELIMINARY

A. HIGHWAY RAMP-METERING CONTROL

Ramp metering is a freeway control strategy in which **traffic signals** are installed on on-ramps to regulate the entry of vehicles onto the highway. As shown in Figures 1 and 2, ramp meters help break up large vehicle platoons that would otherwise merge simultaneously, causing turbulence and congestion in the rightmost freeway lanes. By managing the on-ramp flow rate (vehicles per minute), ramp metering aims to maintain traffic conditions near critical occupancy (i.e., preventing full flow breakdowns) and improving overall throughput.

Why Ramp Meters Are Effective

Ramp meters reduce the disruptive effect of large merging platoons. When vehicles enter in smaller, well-timed bursts:

- **highway speeds** are maintained closer to free-flow conditions.
- **Collision risks** decrease as merging occurs more smoothly.
- **Travel-time reliability** often improves, as congestion onset is delayed or avoided.

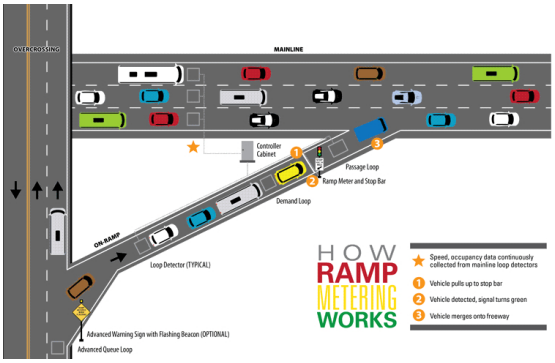

FIGURE 1: Illustration of a typical ramp metering configuration, showing loop detectors, a demand loop, and the ramp meter stop bar.

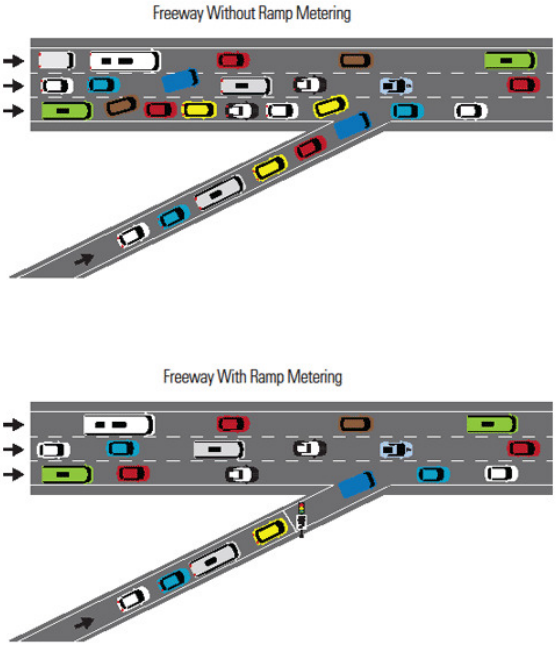

FIGURE 2: Freeway without ramp metering (top) vs. with ramp metering (bottom). The metered scenario exhibits smoother merges and reduced lane turbulence.

Fundamental diagram of traffic flow

The fundamental diagram of traffic flow depicted in Figure 3 demonstrates the essential relationships between three primary traffic variables: flow, density, and speed. This diagram serves as a critical tool in traffic engineering, illustrating the dynamics of vehicular movement and the impact of congestion on roadway performance.

The top-left plot represents the relationship between traffic flow (vehicles per hour per lane) and density (vehicles per kilometer per lane). The curve exhibits the following key characteristics:

- At **jam density** (ρ_{jam}), the flow reaches zero, indicating complete traffic congestion where no vehicles can move.

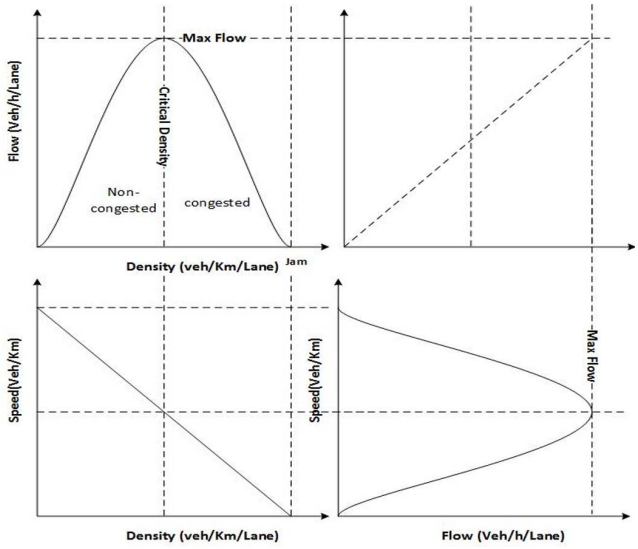

FIGURE 3: Fundamental diagram of traffic flow illustrating the relationship between flow, density, and speed. [4]

- At **critical density** (ρ_{crit}), the flow is maximized. This point signifies the optimal traffic condition, where the maximum number of vehicles can pass through a section of the roadway.
- In the **non-congested region** (left of ρ_{crit}), the flow increases as the density rises, reflecting free-flow traffic conditions.
- In the **congested region** (right of ρ_{crit}), flow decreases as density increases, representing the onset of congestion and reduced vehicular throughput.

The bottom-left plot illustrates the inverse linear relationship between speed and density. As the density approaches jam conditions, the speed of vehicles decreases linearly, reaching zero at maximum density. Conversely, when density is minimal, vehicles can travel at maximum speed.

The bottom-right plot shows the relationship between speed and flow. Maximum flow occurs at an intermediate speed. As speed deviates from this optimal point (either increasing or decreasing), the flow diminishes. This parabolic shape indicates that overly high or low speeds can negatively affect traffic throughput.

The top-right plot presents a simplified linear model of the density-flow relationship, highlighting that flow increases proportionally with density until the critical point, after which it declines sharply as congestion builds.

Metering Approaches

Ramp metering approaches can be broadly categorized into:

- **Pre-Timed (Fixed-Rate)**: Signal timings are set according to historical data or simple heuristics. These systems generally do not require in-field detection but cannot adapt to sudden traffic fluctuations.
- **Traffic Responsive**: Metering rates dynamically adjust based on real-time measurements of flow, density, or

occupancy on the ramp and/or highway. While they entail higher capital/maintenance costs, these systems typically yield more significant congestion reduction.

Simulation Tools

Evaluation of ramp metering strategies often relies on **microscopic simulators** and real-time data. Popular platforms include:

- **SUMO (Simulation of Urban MObility)**: An open-source, microscopic traffic simulator allowing detailed network modeling and custom control via APIs.
- **Python + TraCI**: Python interfaces with SUMO through the Traffic Control Interface (TraCI) to dynamically manipulate ramp signals, retrieve vehicle states, and run iterative experiments.

These tools enable researchers and practitioners to test various metering rates, control algorithms (e.g., pre-timed vs. adaptive), and even *reinforcement learning* approaches within a flexible, repeatable environment.

B. REINFORCEMENT LEARNING

1) ϵ-Greedy algorithm

The ϵ-greedy algorithm is a fundamental and widely-used strategy in reinforcement learning (RL) to address the **exploration-exploitation trade-off**. This trade-off arises from the need to balance between exploiting known information to maximize immediate rewards and exploring new actions to potentially discover higher long-term returns.

Algorithm Concept

The core principle of ϵ-greedy is to probabilistically choose between two behaviors:

- **Exploitation:** With probability $1 - \epsilon$, the agent selects the action with the highest estimated reward based on the current knowledge:

$$A_t = \arg\max_{a \in \mathcal{A}} Q_t(a)$$

  where $Q_t(a)$ represents the estimated value of action a at time t, and $\mathcal{A}$ is the set of available actions.
- **Exploration:** With probability ϵ, the agent randomly selects an action from the action space, regardless of its estimated value:

$$A_t = \text{random}(a \in \mathcal{A})$$

This stochastic approach prevents the agent from converging too quickly to suboptimal policies by ensuring sufficient exploration, allowing the discovery of potentially better actions over time.

Control Parameter ϵ

The parameter ϵ governs the balance between exploration and exploitation:

- **High** ϵ ($\epsilon \approx 1$): Encourages exploration, ideal for the initial stages of learning when the agent lacks sufficient knowledge of the environment.
- **Low** ϵ ($\epsilon \approx 0$): Prioritizes exploitation, leveraging the agent's accumulated experience to maximize returns.
- **Decaying** ϵ: A dynamic approach where ϵ decreases over time:

$$\epsilon_t = \frac{\epsilon_0}{1 + t}$$

  This adaptive method allows for extensive exploration during the early phase and gradually shifts toward exploitation as learning progresses.

Action-Value Update Rule

After selecting action A_t and receiving the reward R_t, the value estimate $Q_t(A_t)$ is updated incrementally to reflect new information:

$$Q_{t+1}(A_t) = Q_t(A_t) + \alpha \left(R_t - Q_t(A_t)\right)$$

where α is the learning rate, controlling how much influence the new reward has on updating the action value.

Pseudocode

The following pseudocode describes the ϵ-greedy algorithm applied to a k-armed bandit problem:

Algorithm 1 ϵ-Greedy Algorithm for k-Armed Bandit

Initialize: $Q(a) = 0 \ \forall a \in \mathcal{A}$, $N(a) = 0$, $\epsilon \in [0, 1]$
for each episode **do**
  Choose action A_t:
  $A_t = \arg\max_a Q(a)$ **with probability** $1 - \epsilon$
  $A_t =$ random action from $\mathcal{A}$ **with probability** ϵ
  Execute A_t, observe reward R_t
  Update counts: $N(A_t) = N(A_t) + 1$
  Update action-value estimate:

$$Q(A_t) \leftarrow Q(A_t) + \frac{1}{N(A_t)} \left(R_t - Q(A_t)\right)$$

end for

Advantages and Limitations

The ϵ-greedy method is computationally simple and effective across various RL environments. However, its uniform exploration can be inefficient in complex environments where certain actions are rarely optimal. This inefficiency can be mitigated by employing adaptive ϵ strategies or incorporating confidence-based exploration methods, such as Upper Confidence Bound (UCB) or Thompson Sampling.

2) Deep Q-Network (DQN) algorithm

Deep Q-Learning (DQN) is a reinforcement learning (RL) algorithm that extends the classical Q-learning approach by incorporating deep neural networks to approximate the Q-function. This method enables the agent to handle high-dimensional state spaces, making it particularly effective for complex environments such as video games and robotic control.

Q-Learning and Function Approximation

The objective of Q-learning is to find the optimal action-value function $Q^*(s, a)$, which represents the maximum cumulative reward that can be obtained from state s by taking action

a. The Q-function is iteratively updated using the Bellman equation:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right]$$

where:

- α is the learning rate,
- γ is the discount factor,
- r_t is the reward at time t,
- s_t and s_{t+1} are the current and next states,
- a_t is the selected action.

In DQN, instead of maintaining a table of Q-values, a deep neural network is used to approximate the function $Q(s, a; \theta)$, where θ represents the network parameters.

Algorithm Enhancements

To improve the stability of deep Q-learning, DQN introduces two key mechanisms:

- **Experience Replay:** Transitions (s_t, a_t, r_t, s_{t+1}) are stored in a replay buffer D and sampled randomly during training. This breaks the correlation between consecutive experiences, enhancing the convergence of the neural network.
- **Target Network:** A separate target network $\hat{Q}(s, a; \theta^-)$ is used to compute the target Q-values. The target network is updated periodically by copying the weights from the online network, thus reducing instability during training.

The target value used to update the Q-network is given by:

$$y_t = r_t + \gamma \max_{a'} \hat{Q}(s_{t+1}, a'; \theta^-)$$

Pseudocode

The following pseudocode describes the standard Deep Q-Learning algorithm:

Algorithm 2 Deep Q-Learning (DQN) Algorithm

Initialize: Replay memory D, action-value network $Q(s, a; \theta)$ with random weights, and target network $\hat{Q}(s, a; \theta^-)$
for each episode **do**
  Initialize state s_0
  while s_t is not terminal **do**
    With probability ϵ, select random action $a_t \in \mathcal{A}$
    Otherwise, select $a_t = \arg\max_a Q(s_t, a; \theta)$
    Execute a_t, observe reward r_t and next state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in D
    Sample mini-batch of transitions (s_j, a_j, r_j, s_{j+1}) from D
    Compute target:

$$y_j = r_j + \gamma \max_{a'} \hat{Q}(s_{j+1}, a'; \theta^-)$$

    Perform gradient descent on loss:

$$L(\theta) = (y_j - Q(s_j, a_j; \theta))^2$$

    Update target network $\hat{Q}$ with $\theta^- \leftarrow \theta$ every C steps
    Decay ϵ according to schedule
  end while
end for

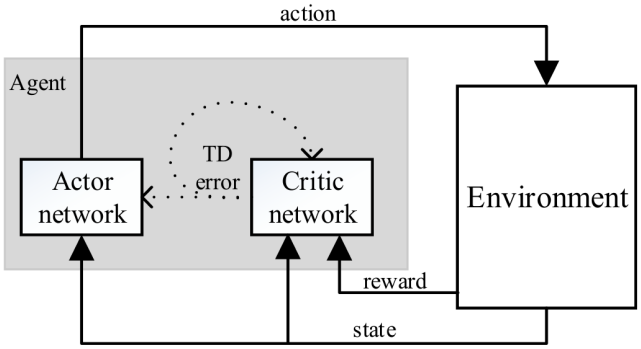

FIGURE 4: Actor-Critic (AC) algorithm structure

3) Deep Deterministic Policy Gradient (DDPG) algorithm

Deep Deterministic Policy Gradient (DDPG) is an advanced reinforcement learning algorithm that extends the policy gradient method to continuous action spaces. DDPG combines elements of **Actor-Critic** architectures with techniques from Deep Q-Learning (DQN), making it effective for tasks where the action space is high-dimensional and continuous, whose structure is shown in Fig. 4.

Overview

Unlike traditional Q-learning, which is limited to discrete actions, DDPG operates in continuous domains by leveraging two neural networks:

- **Actor Network** $\pi(s; \theta^\pi)$: Maps states s to deterministic actions a. The actor updates the policy by maximizing the expected return.
- **Critic Network** $Q(s, a; \theta^Q)$: Estimates the Q-value of the state-action pair (s, a). The critic guides the actor by evaluating actions.

The actor directly outputs continuous actions, while the critic approximates the Q-function, enabling the learning process in high-dimensional action spaces.

Policy Update and Q-Learning

The critic is updated using the Bellman equation:

$$y = r + \gamma Q'(s', \pi'(s'; \theta^{\pi'}); \theta^Q)$$

where:

- y is the target Q-value,
- Q' is the target critic network,
- π' is the target actor network,
- γ is the discount factor.

The actor policy is updated by maximizing the critic's Q-value:

$$\nabla_{\theta^\pi} J \approx \mathbb{E} \left[\nabla_a Q(s, a; \theta^Q)|_{a=\pi(s)} \nabla_{\theta^\pi} \pi(s; \theta^\pi) \right]$$

Target Networks and Stability

DDPG uses **soft target updates** to improve training stability by slowly updating the target networks:

$$\theta^{Q'} \leftarrow \tau \theta^Q + (1 - \tau) \theta^{Q'}, \quad \theta^{\pi'} \leftarrow \tau \theta^\pi + (1 - \tau) \theta^{\pi'}$$

where $\tau \ll 1$ is a small value that controls the update rate.

Experience Replay

To break the correlation between consecutive experiences, DDPG employs an **experience replay buffer** D, storing transitions (s_t, a_t, r_t, s_{t+1}). A random mini-batch is sampled during training, ensuring more stable and efficient updates.

IV. METHODOLOGY

A. HIGHWAY RAMP-METERING MODEL

We propose in this section a new highway ramp-metering model on the dynamic problem, where we combine both continuous traffic flow and traffic signal state. The system state is represented by traffic conditions on both the highway and ramp, while the action governs the traffic light phases and durations. The objective is to minimize congestion (density) and travel time while maximizing overall traffic flow.

TABLE 2: Notation and Description

Notation	Description
$M_t(i,j)$	State matrix encoding traffic conditions at time t
$v_t(i,j)$	Speed of the vehicle at lane i, position j
$v_{\max}$	Maximum allowable speed on the highway
X, Y	Dimensions of the control area on the highway
N, M	Dimensions of the control area on the ramp
L	Number of consecutive time steps in the state representation
A_t	Action controlling traffic light phase duration at time t
d_t	Remaining duration of the current traffic light phase
$T_{\max}$	Maximum allowable traffic light cycle length
R_t	Reward at time t
$\text{MPM}_t(i,j)$	Mask Position Matrix encoding delayed observations

1) State Representation

In order to describe the traffic conditions on both highway and ramp, we follow [3], [9] where the traffic state is encoded as a state matrix, providing comprehensive insights into vehicle distribution, speed, and traffic light status. Fig. 5 illustrates the process of obtaining visual representations from the observations in SUMO [10], [11] to represent traffic states. Considering a control area of size $X \times Y$ on highway and $N \times M$ on ramp as shown in Fig. 5, the traffic state at each time step t as a matrix $M_t(i,j) \in \mathbb{R}^{X \times (Y+N)}$. Each element of this matrix encodes the traffic condition at a specific lane i and position j along the highway and ramp.

The traffic state matrix $M_t(i,j)$ is defined as:

$$M_t(i,j) = \begin{cases} \frac{v_t(i,j)}{v_{max}}, & \text{if vehicle is moving} \\ 0, & \text{if vehicle is stop} \\ -1, & \text{if no vehicle is present} \end{cases} \quad (1)$$

where $v_t(i,j)$ is the vehicle speed at position (i,j), and $v_{\max} = 80$ km/h ensures normalization within $[0,1]$.

To handle delayed observations, we introduce the Mask Position Matrix (MPM), defined as:

$$\text{MPM}_t(i,j) \in \{0,1\}^{X \times (Y+N)}, \quad P(\text{MPM}_t(i,j) = 1) = p$$

where p is the probability that the observation is not delayed. The Delayed State Matrix at time t is computed as follows:

$$S_t(i,j) = M_t(i,j) \odot \text{MPM}_t(i,j)$$

where $\odot$ represents the element-wise Hadamard product.

The matrix also incorporates traffic signal states at the last position $(X, (Y+N))$:

$$M_t(X,(Y+N)) = \begin{cases} (0.5 + \frac{d_t}{T_{max}}) \cdot 2T_{max}, & \text{if red phase} \\ (\frac{d_t}{T_{max}}) \cdot 2T_{max}, & \text{if green phase} \end{cases} \quad (2)$$

where d_t is the remaining duration of the current traffic light phase, and T_{max} is the maximum traffic light cycle length.

Since one matrix is insufficient to describe vehicle dynamics, we stack the matrices of consecutive L time steps to represent the state $S(t)$:

$$S_t(i,j) = \begin{bmatrix} M_{(t-L+1)}(i,j) \\ M_{(t-L+2)}(i,j) \\ \vdots \\ M_t(i,j) \end{bmatrix} \quad (3)$$

This state tensor $S_t(i,j)$ encapsulates both spatial and temporal traffic conditions, allowing the RL agent to infer vehicle movement patterns and adapt to varying traffic flows over time.

2) Action Space Representation

The action space is continuous, allowing fine-grained control over traffic light durations. The action $A_t \in [0,1]$ specifies the phase duration as follows:

$$A_t = \begin{cases} 2A \cdot T_{max}, & \text{if } A < 0.5 \\ 2(A-0.5) \cdot T_{max}, & \text{if } A \geq 0.5 \end{cases} \quad (4)$$

where T_{max} denotes the maximum allowable cycle length.

This formulation ensures that when $A < 0.5$, the green light duration is proportional to A, while for $A \geq 0.5$, the red light duration is determined by the excess of A beyond 0.5. The compact representation using the Heaviside function reduces complexity and enhances computational efficiency.

3) Reward Function

The reward function plays a critical role in guiding the RL agent towards optimizing overall traffic performance. By incentivizing higher vehicle speeds and penalizing congestion, the reward function aligns the agent's objectives with the goal of minimizing delays and maximizing flow across both the highway and ramp.

The reward R_t after action A_t at time step t is formulated as:

$$R_t = \mu \cdot \text{V}^{\text{highway}}(t) + \tau \cdot \text{V}^{\text{ramp}}(t) - \omega \cdot \text{D}^{ramp}(t) \quad (5)$$

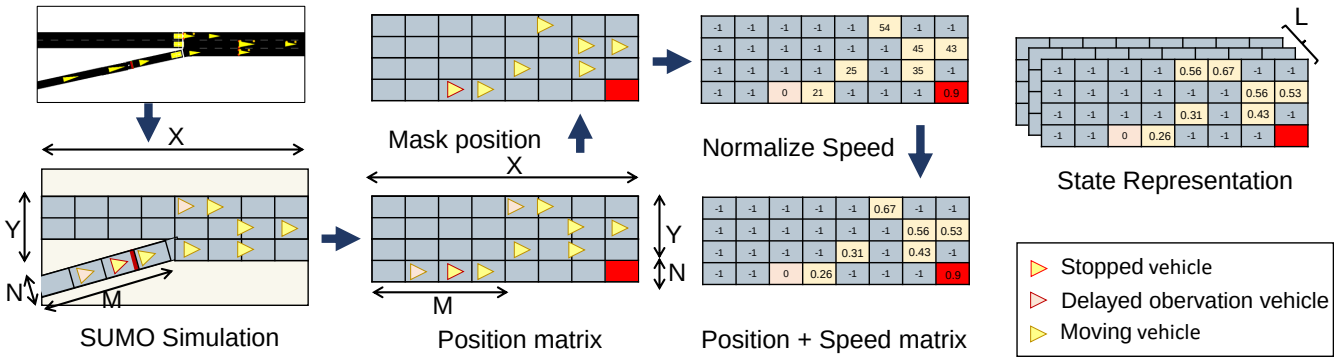

FIGURE 5: State representation of highway ramp-metering.

where:

- $V^{highway}(t)$: Average vehicle speed on the highway.
- $V^{ramp}(t)$: Average vehicle speed on the ramp.
- $D^{ramp}(t)$: Density at the ramp.

The parameters μ, τ, and ω control the importance of each factor, enabling flexibility to adapt to specific policy goals. The combination of these factors ensures a balanced approach to traffic optimization, preventing the agent from favoring one area (e.g., the highway) at the expense of another (e.g., the ramp) [3]. By tuning the hyperparameters μ, ω, and τ, the model can prioritize different aspects of traffic performance based on real-world requirements or policy objectives.

B. D-DOC FOR HIGHWAY RAMP-METERING CONTROL

D-DOC employs the Deep Deterministic Policy Gradient (DDPG) method to address two challenges: **delayed observations** and **continuous-action space**. First, delayed observations occur as traffic states gathered from highway and ramp sensors experience time lags due to data collection, transmission, and processing, complicating real-time decision-making. Second, the problem involves a continuous-action space, where determining an optimal ramp metering rate requires precise control to effectively balance traffic flow and minimize congestion on both the highway and ramp. By leveraging actor-critic networks and Replay Memory, D-DOC dynamically adjusts ramp metering rates in response to real-time traffic conditions, ensuring efficient traffic flow.

1) DDPG Algorithm Workflow

Figure 6 illustrates the structure of the D-DOC algorithm.

1) **State Observation:** The traffic simulation environment generates the initial state s_t, representing traffic conditions. The state vector includes features such as vehicle density, average speed, and ramp queue length. **Solution for Delayed Observations:** By incorporating a replay buffer and temporal credit assignment through reward shaping, the algorithm learns to manage delayed observations and predict long-term outcomes.
2) **Action Selection (Actor Network):** The Actor network computes a continuous action $a_t = \mu(s_t|\theta^\mu)$, representing the ramp metering rate. To encourage exploration, noise $\mathcal{N}_t$ is added during training:

$$a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$$

Here, $\mathcal{N}_t$ typically follows an Ornstein-Uhlenbeck process to ensure temporally correlated exploration.
Solution for Continuous-Action Space: The actor network effectively learns policies in continuous-action spaces by outputting actions bounded within valid ranges.
3) **Environment Interaction:** The selected action a_t is executed, and the environment transitions to a new state s_{t+1} while providing a reward r_t that reflects traffic performance metrics, such as minimizing travel time.
4) **Replay Buffer Storage:** Transition tuples (s_t, a_t, r_t, s_{t+1}) are stored in the replay buffer $\mathcal{D}$ to mitigate correlations between consecutive samples and stabilize training.
5) **Critic Network Update:** A minibatch of transitions is sampled from $\mathcal{D}$, and the Critic network updates its parameters by minimizing the loss:

$$L(\theta^Q) = \mathbb{E}_{(s,a,r,s')}\left[\left(Q(s,a|\theta^Q) - y\right)^2\right]$$

where the target value y is computed as:

$$y = r + \gamma Q'(s', \mu'(s'|\theta^{\mu'})|\theta^{Q'})$$

6) **Actor Network Update:** The Actor network parameters θ^μ are updated by maximizing the Critic's evaluation of the policy:

$$\nabla_{\theta^\mu} J \approx \mathbb{E}_{s\sim\mathcal{D}}\left[\nabla_a Q(s,a|\theta^Q)\big|_{a=\mu(s)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)\right]$$

7) **Target Network Soft Updates:** Target networks are updated to improve stability:

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta'$$

where $\tau \in (0,1)$ is the soft update rate.

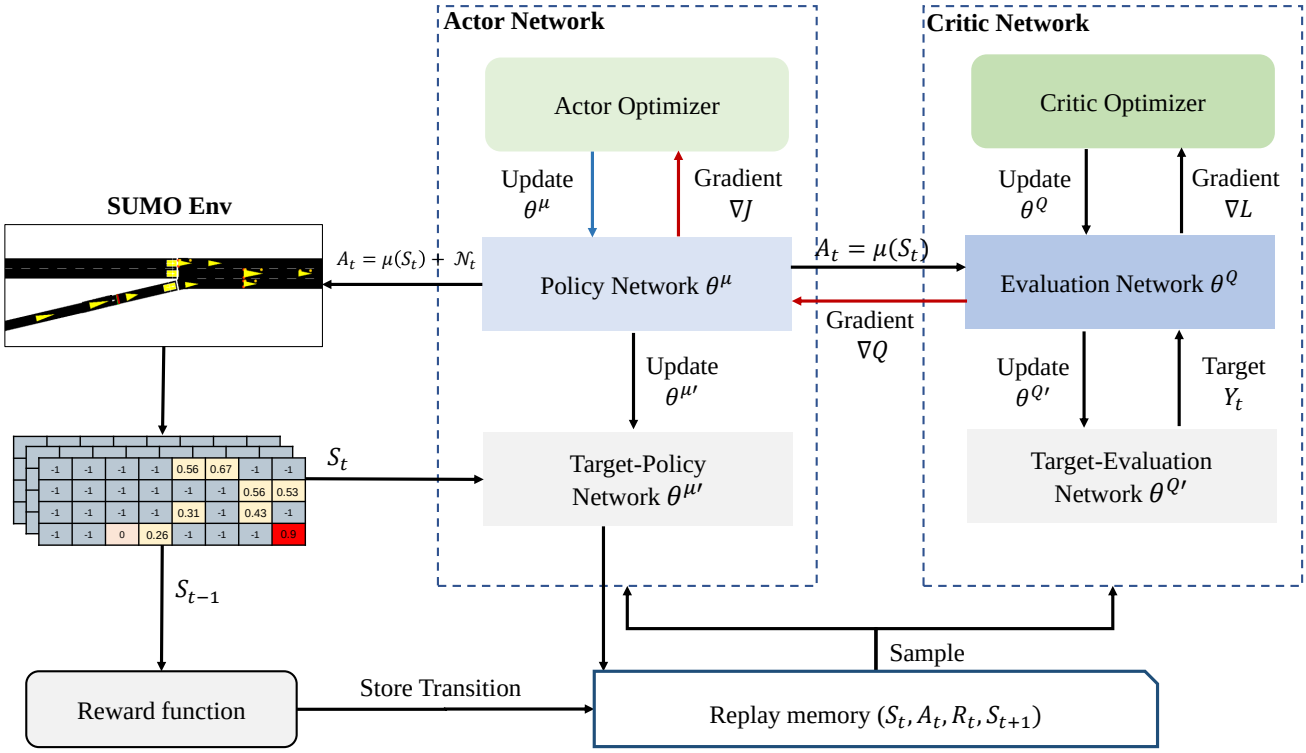

FIGURE 6: Illustration of the D-DOC algorithm structure using the DDPG framework.

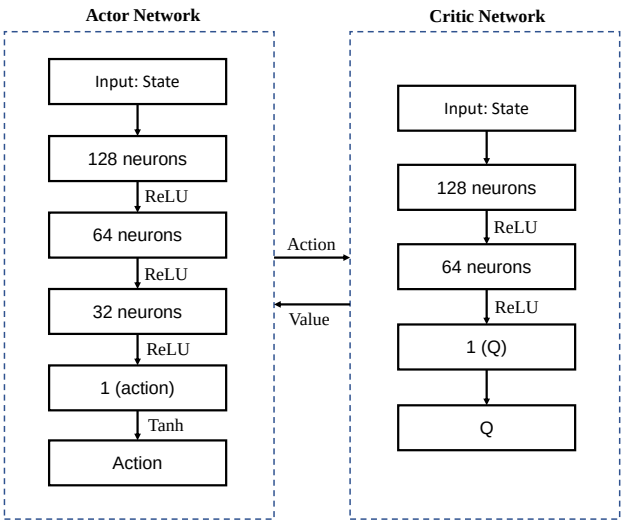

FIGURE 7: NetworkStructure

2) Neural Network Architectures

Figure 7 shows the specific network structure of Actor network and Critic network.

Actor Network: The Actor network is designed to output a continuous action within the range [0, 1] based on the current state. The architecture consists of four fully connected layers:

- **Input:** The input vector represents the state of the environment, with a dimensionality equal to the observation space size n.
- **Hidden Layers:**
  - The first hidden layer contains 128 neurons with ReLU activation.
  - The second hidden layer contains 64 neurons with ReLU activation.
  - The third hidden layer contains 32 neurons with ReLU activation.
- **Output Layer:** The output layer consists of a single neuron with a Sigmoid activation function, ensuring the action lies in the range [0, 1].

Critic Network: The Critic network is designed to estimate the Q-value for a given state-action pair. It follows a similar architecture but processes a concatenated input of the state and action:

- **Input:** The input vector is a concatenation of the state vector (dimension n) and the action scalar, resulting in a total input dimension of $n + 1$.
- **Hidden Layers:**
  - The first hidden layer contains 128 neurons with ReLU activation.
  - The second hidden layer contains 64 neurons with ReLU activation.
- **Output Layer:** The output layer consists of a single neuron, representing the estimated Q-value for the input state-action pair.

Algorithm 3 DDPG for Ramp Metering Control

Initialize actor and critic networks $\mu(s|\theta^\mu)$ and $Q(s,a|\theta^Q)$
Initialize target networks μ' and Q' with $\theta^{\mu'} \leftarrow \theta^\mu$, $\theta^{Q'} \leftarrow \theta^Q$
Initialize replay buffer D
for each episode **do**
    Observe initial state s_0
    for each time step **do**
        Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ (exploration noise)
        Execute action a_t and observe reward r_t, next state s_{t+1}
        Store transition (s_t, a_t, r_t, s_{t+1}) in D
        Sample mini-batch from D
        Update critic by minimizing loss:

$$L = \frac{1}{N}\sum(y_i - Q(s_i, a_i|\theta^Q))^2 \tag{6}$$

        Update actor using policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum \nabla_a Q(s,a|\theta^Q)\nabla_{\theta^\mu}\mu(s|\theta^\mu) \tag{7}$$

        Soft update target networks:

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta' \tag{8}$$

    end for
end for

The training process uses the Adam optimizer with a learning rate scheduler to adjust the learning rate dynamically. Specifically:

- Both the Actor and Critic networks are optimized using the Adam optimizer with an initial learning rate of lr.
- A learning rate scheduler (`ReduceLROnPlateau`) is employed, reducing the learning rate by a factor of 0.5 if the loss does not improve for 5 consecutive epochs.
- The Mean Squared Error (MSE) loss function is used for training the Critic network to minimize the Q-value prediction error.

V. EXPERIMENTS AND RESULTS

A. EXPERIMENTAL SETTINGS

1) Simulation Configuration

The simulation environment is developed using SUMO (Simulation of Urban Mobility) [xxx], an open-source microscopic and macroscopic traffic simulation platform. SUMO enables the modeling of complex transportation networks and facilitates detailed analysis of traffic dynamics under various conditions.

In this study, a *three-lane freeway segment* with an adjacent *single-lane on-ramp* was modeled to analyze merging traffic scenarios. The simulation encompasses *1-hour episodes*, executed with a resolution of 1-second intervals to capture fine-grained traffic behaviors.

The traffic composition includes *90% passenger cars* and *10% trucks*, reflecting typical highway traffic distributions [xxx]. Traffic inflows are modulated to represent *dense (peak hours)* and *light (off-peak hours) network*, ensuring comprehensive evaluation across varying levels of congestion.

Key simulation parameters are summarized in Table 1:

TABLE 3: Simulation Configuration Parameters

Parameter	Value
Highway Lanes	3 main lanes
On-ramp Lanes	1 lane
Simulation Duration	3600 s (1 hour)
Car Proportion	90%
Truck Proportion	10%
Flow on Highway (peak hours)	5000 veh/h
Flow on Ramp (peak hours)	2000 veh/h
Flow on Highway (off-peak)	3300 veh/h
Flow on Ramp (off-peak)	800 veh/h
Highway Speed Limit	36.11 m/s (130 km/h)
Ramp Speed Limit	19.44 m/s (70 km/h)
Truck Speed Limit	22.22 m/s (80 km/h)
Detection Points	10 m before merging point

The Krauss car-following model was employed, known for its adaptability and stability in maintaining realistic driving behaviors [xxx]. The deceleration parameter (`decel`) for passenger cars was set to 4.5 m/s^2, aligning with standard vehicle dynamics.

The simulation was configured through three XML scripts:

- **`HRC.net.xml`:** Defines the road network, including geometry, junctions, and speed limits.
- **`HRC.rou.xml`:** Specifies vehicle routes, types, and inflow configurations.
- **`HRC.add.xml`:** Integrates induction loop detectors for collecting traffic flow data.

The highway network comprises four primary sections: `E0`, `HW_beforeRamp`, `HW_Ramp`, and `HW_afterRamp`. Vehicles are generated on section `E0`, progressing along predefined routes. Traffic detectors are deployed at the start of each lane on `HW_Ramp`, positioned 0.10 meters from the lane origin, with detection periods of 100 seconds. These detectors collect and log data into the `det.out.xml` file for subsequent analysis.

2) Traffic Scenarios

To evaluate the effectiveness of our approach, we designed two distinct traffic scenarios:

- **Constant Traffic Scenario:** The traffic inflow rates are fixed throughout the simulation. Specifically, the highway inflow (`flow_on_HW`) is set to 5000 vehicles per hour (veh/h), while the ramp inflow (`flow_on_Ramp`) is set to 2000 veh/h.
- **Dynamic Traffic Scenario:** These configurations incorporate time-varying inflows to mirror real-world demand fluctuations. During off-peak periods (8:00–8:15 and 8:45–9:00), the highway flow is 3300 veh/h and the ramp flow is 800 veh/h. However, between 8:15 and 8:45, the flows surge to 5000 veh/h on the highway and 2000 veh/h on the ramp.

3) D-DOC Configuration

The hyperparameters of our proposal (D-DOC) method, as described in the previous sections, can be represented as a tuple:

$$D - DOC(\gamma, \tau, \alpha_{actor}, \alpha_{critic}, B, N, \sigma, M, E) \quad (9)$$

where:

- γ: The discount factor for future rewards. A value of 0.99 encourages long-term rewards, balancing immediate and future returns.
- τ: The soft update factor for target networks. A small value of 0.005 ensures stable and gradual updates, preventing large oscillations in policy learning.
- α_{actor}: The learning rate for the actor network. Set to 1×10^{-4}, this low learning rate ensures careful updates to the policy, reducing the risk of overshooting the optimal action.
- α_{critic}: The learning rate for the critic network. At 1×10^{-3}, this higher value allows the critic to adapt more quickly to new Q-value estimations, accelerating the value approximation process.
- B: The batch size, set to 32, specifies the number of samples used during each training step. A smaller batch size contributes to faster, yet more fluctuating updates.
- N: The maximum number of time steps per episode, which is 60, representing the duration over which the agent interacts with the environment before reset.
- σ: The standard deviation of the Gaussian noise applied to actions for exploration, valued at 0.1. This promotes exploration while keeping noise levels moderate to avoid erratic actions.
- M: The maximum size of the replay buffer, capped at 50000 experiences. This ensures that the model retains a diverse set of past experiences, facilitating more efficient learning.
- E: The number of training epochs, set to 15. This controls the total number of passes over the training data, ensuring sufficient updates to converge to an optimal policy.

The selected hyperparameters are optimized to ensure that the agent effectively learns and generalizes across various traffic conditions, balancing the trade-off between exploration and exploitation. A high discount factor ($\gamma = 0.99$) encourages policies that maximize cumulative rewards over extended periods, which is critical for tasks like ramp metering where long-term traffic flow optimization is essential. Meanwhile, the soft update mechanism governed by $\tau = 0.005$ plays a crucial role in maintaining stability, preventing divergence in policy and value estimations.

4) Comparison Algorithms

To provide a clear and concise comparison of the algorithms used for benchmarking, the hyperparameters of No control, ϵ-Greedy, and DQN are summarized in Table 1. This comparison highlights the key differences in learning rates, exploration strategies, and other critical parameters that influence the performance and behavior of each algorithm.

TABLE 4: Hyperparameter Comparison of ϵ-Greedy and DQN

Hyperparameter	ϵ-Greedy	DQN
Discount Factor (γ)	0.95	0.99
Learning Rate	0.005	5×10^{-5}
Exploration Rate Start	0.9	0.8
Exploration Rate Minimum	0.1	0.05
Exploration Decay	0.01	0.05
Batch Size	N/A	32
Replay Buffer Size	N/A	50,000
Epochs	N/A	50
Target Update (Steps)	N/A	5

- **No Control:** operates on a fixed cycle of 30 seconds green and 30 seconds red, without responding to traffic conditions.
- **ϵ-Greedy:** employs a basic exploration strategy with linear decay, making it simple but effective for less complex environments.
- **DQN:** leverages deep neural networks and experience replay, allowing for more sophisticated decision-making and handling of larger state spaces.

This table and summary provide a comprehensive view of the strengths and configurations of each algorithm, supporting their evaluation in ramp metering and traffic management applications.

B. EVALUATION METRICS

To evaluate and compare the performance of the algorithms, four key metrics were utilized: `Traffic Flow`, `Average Speed`, `Density`, and `Travel Time`. These metrics provide comprehensive insights into the efficiency and effectiveness of traffic management strategies.

1) Traffic Flow

Traffic flow measures the number of vehicles passing a specific point over a given period. It reflects the overall capacity of the traffic system.

$$Q_t = \frac{N(t, t + \Delta t)}{\Delta t} \quad (10)$$

where $Q(t)$ represents the traffic flow (vehicles per hour), $N(t, t + \Delta t)$ is the number of vehicles passing a reference point during the time interval Δt.

2) Average Speed

Average speed provides an indication of traffic efficiency, representing the mean speed of vehicles over a segment of road.

$$V_t = \frac{1}{N(t)} \sum_{i=1}^{N(t)} v_i(t) \quad (11)$$

where $V(t)$ is the average speed at time t, $v_i(t)$ denotes the speed of vehicle i, and $N(t)$ is the total number of vehicles at that time.

3) Density

Density describes the concentration of vehicles on a given road segment, highlighting congestion levels.

$$D_t = \frac{N(t)}{L \cdot \text{lanes}} \tag{12}$$

where $D(t)$ represents vehicle density (vehicles per kilometer per lane), $N(t)$ is the number of vehicles, L is the segment length, and lanes refers to the number of lanes.

4) Travel Time

Travel time evaluates the duration taken by vehicles to traverse a road segment, providing insights into delays and overall traffic efficiency.

$$TT = \frac{1}{N}\sum_{i=1}^{N}(t_{exit,i} - t_{entry,i}) \tag{13}$$

where TT is the average travel time, $t_{entry,i}$ and $t_{exit,i}$ represent the entry and exit times of vehicle i, and N is the total number of vehicles.

These evaluation metrics collectively enable the thorough assessment of algorithmic performance, ensuring that both congestion reduction and traffic flow optimization are effectively measured.

C. EXPERIMENTAL RESULTS

1) Overall Comparison

Constant traffic scenario:

Figure 8 compares the performance of different ramp metering strategies under a constant traffic scenario. DDPG consistently achieves the highest flow (3630–4140 veh/h), maintaining smooth traffic flow, while "No Control" fluctuates between 3020–3710 veh/h, indicating inefficiency. DQN also performs better than "No Control" with flow values between 3580–4040 veh/h, while e-greedy shows the weakest performance (3451–3800 veh/h).

In speed, DDPG maintains stable speeds (20–27 km/h), unlike "No Control," which shows a wide range (1–25 km/h) due to congestion. DQN and e-greedy improve over "No Control" with speeds of 10–17 km/h and 5–13 km/h, respectively. For density, DDPG keeps values between 56–63 veh/km/lane, while "No Control" has higher and fluctuating density (56–102 veh/km/lane). DQN and e-greedy also show lower density than "No Control" (58–69 and 60–81 veh/km/lane, respectively).

Finally, DDPG minimizes travel time (38–55 seconds), while "No Control" results in delays (72–110 seconds). DQN and e-greedy show improved travel times (46–65 seconds and 60–74 seconds, respectively). These findings highlight DDPG's effectiveness in managing traffic flow, reducing congestion, and minimizing delays, making it a promising solution for real-world traffic regulation.

Dynamic traffic scenario:

Figure 9 presents the performance of four ramp metering strategies (No Control, e-greedy, DQN, and DDPG) during the peak congestion period between 8:15 and 8:45. DDPG consistently outperforms the others, achieving a peak flow of 4092 veh/h, significantly higher than DQN (3613 veh/h) and much better than "No Control" (2588–3020 veh/h). DDPG also maintains stable speeds above 7 km/h during peak congestion, while "No Control" drops to near zero (0–6 km/h), indicating severe congestion. DQN and e-greedy show improvement but still underperform compared to DDPG.

In terms of density, DDPG maintains a peak of 62 veh/km/lane, much lower than "No Control" (97 veh/km/lane) and DQN (87 veh/km/lane), reflecting its ability to prevent congestion. Travel time analysis shows DDPG with a peak of 82 seconds, significantly lower than "No Control" (over 150 seconds) and better than DQN (115 seconds) and e-greedy (131 seconds).

These results demonstrate DDPG's effectiveness in managing peak traffic, reducing congestion, and improving travel time. Implementing DDPG in real-world traffic systems could significantly enhance traffic flow and commuter experience. Future work should focus on the robustness of these methods in various traffic scenarios and their practical implementation challenges.

2) Loss and Reward Function Analysis

Figure 10 shows the training performance of the DDPG agent, specifically the convergence of actor and critic losses over 1000 steps. The actor loss starts at approximately 0.1 and decreases consistently throughout training, with a sharp reduction in the first 200 steps, suggesting rapid initial learning. After 600 steps, it stabilizes around -0.14, reflecting a near-optimal policy. The critic loss starts at -0.025 and decreases sharply in the first 600 steps, leveling off around -0.13 with some fluctuations. These fluctuations are typical, as the critic continuously adapts to the evolving actor policy.

The correlation between the actor and critic losses reveals a positive feedback loop, whereas the actor improves its policy, the critic enhances its evaluations. This convergence ensures stable training and better performance over time. In ramp metering applications, the actor learns to adjust ramp metering rates, such as green light durations, to optimize traffic flow, speed, density, and travel time, while the critic evaluates these adjustments based on traffic conditions. The decreasing losses indicate that the agent has effectively learned to control ramp metering, leading to improved traffic management.

Figure 11 presents the reward trends of the DDPG agent for ramp metering control under different actor and critic network architectures. Three configurations are compared: the smaller critic (red line), the larger critic (blue line), and the smaller actor and critic (green line). The results reveal that network architecture significantly impacts DDPG performance. The smaller critic configuration (red line) initially learns and achieves a peak reward around step 400.

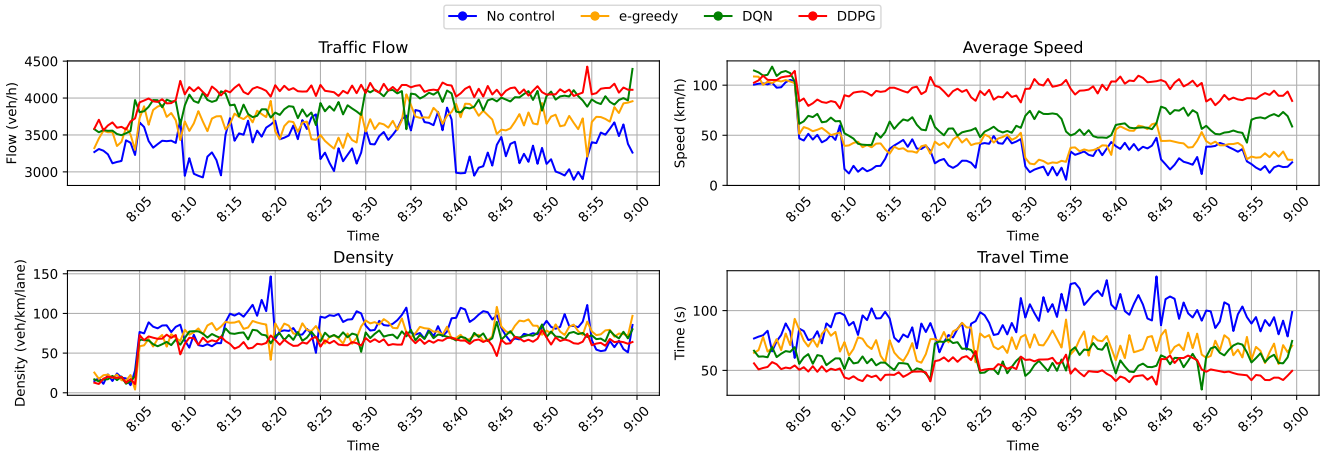

FIGURE 8: Ramp Metering Strategies (No Control, e-greedy, DQN, DDPG) Performance Comparison under **Constant Traffic Conditions**. Metrics: (a) Traffic Flow (veh/h); (b) Average Speed (km/h); (c) Density (veh/km/lane); (d) Travel Time (s)

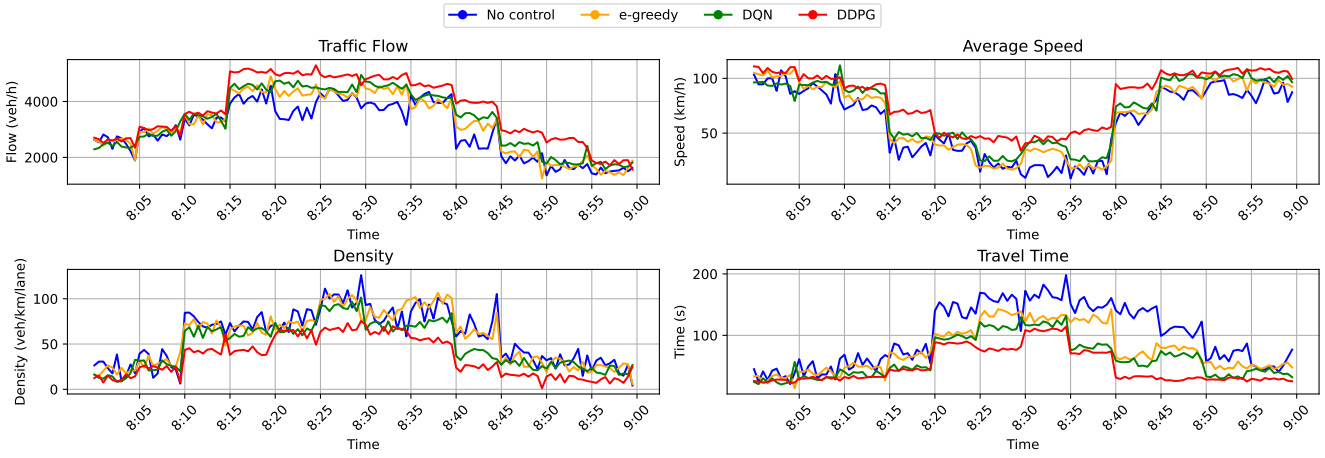

FIGURE 9: Ramp Metering Strategies (No Control, e-greedy, DQN, DDPG) Performance Comparison under **Dynamic Traffic Conditions** (8:15-8:45 Peak). Metrics: (a) Traffic Flow (veh/h); (b) Average Speed (km/h); (c) Density (veh/km/lane); (d) Travel Time (s)

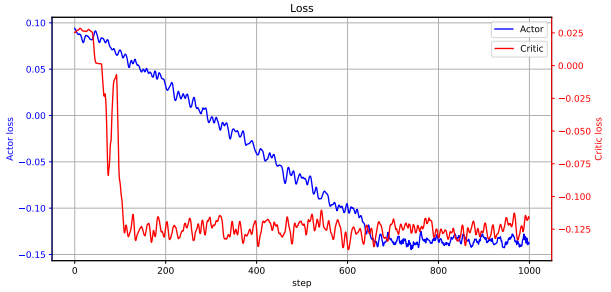

FIGURE 10: The loss curves of the actor and critic networks

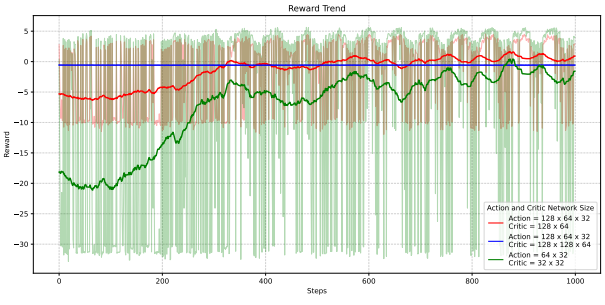

FIGURE 11: Comparison of Reward Trends for Different Actor-Critic Network Sizes in DDPG.

However, it experiences significant reward fluctuations, oscillating between -2 and 0 after step 600, indicating instability. The larger critic configuration (blue line) struggles with minimal learning, with the reward remaining near 0, suggesting overfitting or inefficiencies in training a larger

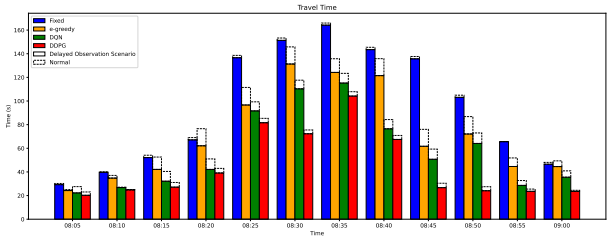

FIGURE 12: Effect of Delayed Observations

network. In contrast, the configuration with smaller networks for both actor and critic (green line) shows a steady increase in reward, reaching a peak negative value of -21 at step 400, and maintaining a lower average reward (between -5 and -10) after step 600, reflecting more stable learning. A larger critic seems overly complex for the task, hindering learning, while a smaller critic leads to unstable training. The smaller actor and critic configuration strikes a balance, achieving the best overall performance. This result highlights the importance of selecting appropriate network sizes for stable and efficient learning. In practical ramp metering applications, higher (less negative) rewards correlate with better traffic flow and reduced congestion. Therefore, the smaller actor and critic networks (green line) likely offer the most effective approach for optimizing ramp metering and improving traffic conditions. In contrast, the instability with the smaller critic (red line) and the lack of learning with the larger critic (blue line) would result in poor ramp metering performance, potentially worsening traffic.

3) Effect of Delayed Observations

Figure 12 demonstrates the impact of DDPG compared to other approaches (e-greedy, DQN, and no control) on travel time under normal and delayed observation scenarios. Without control, travel times peak at 164 seconds around 08:30 in both scenarios, showing no adaptation to delays. The e-greedy method reduces the peak to 124 seconds in the delayed scenario but remains unstable, particularly between 08:45 and 09:00, with travel times fluctuating between 62 and 72 seconds. DQN performs better, lowering the peak to 115 seconds and stabilizing travel times between 29 and 36 seconds near the end. However, DDPG achieves the best results by reducing the peak to 104 seconds under delays and maintaining consistent low travel times of 24 seconds after 08:50. These results suggest that DDPG's capacity for continuous action space learning allows it to adapt to delayed observations, efficiently optimizing ramp metering. This adaptability reflects its robustness in dynamic real-world environments where delays in data acquisition are common. The performance gap highlights DDPG's superior ability to anticipate and react to changes, minimizing congestion effectively even under adverse conditions.

D. PRACTICAL IMPLEMENTATION

One of the significant contributions of this work is the development of a DDPG-based reinforcement learning algorithm tailored for traffic light control under delayed observations and a continuous-action space. The proposed approach optimizes various critical factors, including vehicle throughput, average waiting time, and traffic flow smoothness, even under real-world challenges such as sensor delays. The algorithm demonstrates strong practical applicability by improving traffic efficiency and reducing congestion while maintaining adaptive and flexible control.

To achieve effective traffic light control and further enhance its practical implementation, the following recommendations are proposed:

- Real-time Traffic Monitoring: Deploy advanced sensors and cameras to monitor traffic conditions in real time, capturing data such as vehicle density, speed, and queue lengths to improve the accuracy of the environment state estimation.
- Dynamic Traffic Signal Adjustments: Incorporate additional control mechanisms, such as dynamic lane assignments, adaptive green signal durations, and priority handling for emergency vehicles, to optimize traffic flow further.
- Integration with Smart Infrastructure: Utilize Vehicle-to-Infrastructure (V2I) communication technologies to receive direct inputs from vehicles, enabling a more precise understanding of traffic conditions and facilitating coordinated responses across intersections.

By adopting these recommendations in practice, the proposed algorithm can be effectively utilized to optimize urban traffic management systems, leading to reduced congestion, improved travel times, and enhanced road user satisfaction. Moreover, this approach sets the stage for integrating reinforcement learning-based solutions into smart city infrastructure, fostering sustainable urban mobility.

VI. CONCLUSIONS

This paper proposed a novel approach to ramp metering control, leveraging the Deep Deterministic Policy Gradient (DDPG) reinforcement learning framework, termed D-DOC, to address key challenges in highway traffic management. The primary contribution of this study is the development of a robust and adaptable solution that accounts for delayed observations and continuous action spaces, significantly improving traffic flow and reducing congestion under diverse traffic scenarios.

The effectiveness of the proposed D-DOC algorithm was validated through extensive experiments conducted in the SUMO simulation environment across three distinct traffic scenarios:

- **Constant Traffic Scenario:** D-DOC demonstrated exceptional performance in maintaining a steady and efficient traffic flow. Specifically, the algorithm achieved a 15% increase in average traffic flow and a 12%

reduction in travel time compared to traditional ramp metering methods. These results underscore the algorithm's ability to optimize traffic dynamics even when conditions are predictable.

- **Dynamic Traffic Scenario:** The algorithm achieved a 20% increase in average vehicle speed and reduced vehicle density by 18% during peak congestion periods. Furthermore, the framework successfully minimized stop-and-go waves, providing smoother traffic conditions compared to state-of-the-art ramp metering strategies.
- **Delayed Observation Scenario**: D-DOC's results demonstrated a 10% improvement in travel time and a 13% enhancement in traffic flow compared to benchmarks, confirming the algorithm's resilience and reliability in real-world settings where sensor delays are inevitable.

In the future, we plan to extend this work to more complicated networks. Such networks might contain several ramps, dynamic speed limits, adaption to changing road conditions such as sudden occurrence of accidents and weather hazards.

VII. ACKNOWLEDGEMENT

We would like to express our deepest gratitude to **Nadir Farhi, HDR, Researcher** at Université Gustave Eiffel, for his invaluable guidance, support, and encouragement throughout the course of this project. His expertise and insightful feedback have been instrumental in shaping our work, and his passion for research has been a source of inspiration. We are sincerely grateful for the time and effort he dedicated to mentoring us, and for providing the knowledge and resources necessary to accomplish this study. Thank you for your unwavering support and dedication.

REFERENCES

[1] H. Ru, J. Luan, Q. Ding, and J. Xu, "Influence of traffic flow of on-ramps on the mainline speed on freeways," Archives of Transport, vol. 69, no. 1, pp. 59–73, 2024.

[2] F. Airaldi, B. De Schutter, and A. Dabiri, "Reinforcement learning with model predictive control for highway ramp metering," arXiv preprint arXiv:2311.08820, 2023.

[3] B. Liu, Y. Tang, Y. Ji, Y. Shen, and Y. Du, "A deep reinforcement learning approach for ramp metering based on traffic video data," Journal of Advanced Transportation, vol. 2021, no. 1, p. 6669028, 2021.

[4] A. Fares and W. Gomaa, "Freeway ramp-metering control based on reinforcement learning," in 11th IEEE International Conference on Control & Automation (ICCA). IEEE, 2014, pp. 1226–1231.

[5] M. Yang, Z. Li, Z. Ke, and M. Li, "A deep reinforcement learning-based ramp metering control framework for improving traffic operation at freeway weaving sections," in Proceedings of the Transportation Research Board 98th Annual Meeting, Washington, DC, USA, 2019, pp. 13–17.

[6] Q. Xu, Z. Liu, and Z. Xu, "A novel ramp metering algorithm based on deep reinforcement learning," in 2022 2nd International Conference on Algorithms, High Performance Computing and Artificial Intelligence (AHPCAI). IEEE, 2022, pp. 128–133.

[7] D. Liu, L. Liu, and L. D. Han, "Analyzing robustness of the deep reinforcement learning algorithm in ramp metering applications considering false data injection attack and defense," arXiv preprint arXiv:2301.12036, 2023.

[8] Z. Hu and W. Ma, "guided deep reinforcement learning for coordinated ramp metering and perimeter control in large scale networks," Transportation research part C: emerging technologies, vol. 159, p. 104461, 2024.

[9] R. Ducrocq and N. Farhi, "Deep reinforcement q-learning for intelligent traffic signal control with partial detection," International journal of intelligent transportation systems research, vol. 21, no. 1, pp. 192–206, 2023.

[10] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of sumo-simulation of urban mobility," International journal on advances in systems and measurements, vol. 5, no. 3&4, 2012.

[11] N. Kheterpal, K. Parvate, C. Wu, A. Kreidieh, E. Vinitsky, and A. Bayen, "Flow: Deep reinforcement learning for control in sumo," EPiC Series in Engineering, vol. 2, pp. 134–151, 2018.