5G NR Codes and Modulation Deep-RL Optimization for uRLLC in Vehicular OCC

Amirul Islam, Nikolaos Thomos, and Leila Musavian

***Abstract*—In dynamic and time-varying vehicular networks, existing vehicular communication systems cannot guarantee ultra-reliable and low latency communication (uRLLC). To address this, we propose a novel deep reinforcement learning-based vehicular optical camera communication (OCC) system with an aim to maximize the throughput and ensure uRLLC. To achieve this, our scheme chooses the optimal code rate, modulation scheme and speed of vehicles for multiple vehicular links. We use OCC, which offers interference-free communication as an alternative to radio frequency systems. Moreover, we employ 5G New Radio low-density parity-check codes and an adaptive modulation scheme to support variable rates and ultra-reliability. The proposed large-scale and continuous problem is solved through an actor-critic algorithm based on Wolpertinger architecture. We extendedly evaluate the system performance and compare it with several other schemes from the literature as well as with variants of our scheme. We observe from the results that the proposed method achieves higher average throughput and lower latency than all the other schemes under comparison. Further, the proposed scheme can meet the uRLLC constraints, whereas other schemes under comparison fail to respect these constraints most of the time.**

***Index Terms*—Deep reinforcement learning, vehicular OCC, uRLLC, LDPC codes, multi-agent system.**

I. Introduction

Inter-vehicular communication has been attracting the interest of academia and industry, where autonomous vehicles (AVs) communication will play an essential role for intelligent transportation systems (ITSs) [1]. AV communications are time-varying and highly dynamic, where data must be delivered reliably within stringent time constraints to ensure safety. This makes it challenging to respect ultra-reliable and low-latency communication (uRLLC) in vehicular networks. Various candidate solutions exist, such as radio frequency (RF) technology, e.g., cellular, Wi-Fi, and sensor networks [2], [3]. However, the lack of effective and well-established technologies to meet uRLLC and mitigate interference necessitates the development of a new alternative solution in wireless communication. Optical camera communication (OCC) has emerged as a potential technology for ITS [4], [5] and as an alternative to RF due to the fact it offers license-free unlimited spectrum, lower power consumption, lower implementation cost, longer lifespans, and enhanced security [4]. OCC uses light-emitting diodes (LEDs) as the transmitter and cameras as the receiver [4].

Although OCC offers several advantages, it cannot match the reliability requirements of uRLLC, which is why channel codes should be utilized in OCC systems. Convolutional, Turbo, low-density parity-check (LDPC), and Polar codes are commonly used channel coding schemes [6]. Polar codes are optimal, but this comes at the cost of high complexity. This prohibits their application in real-time systems, while their complexity renders them inappropriate for practical systems. In this work, we use LDPC codes, which are part of fifth generation (5G) new radio (NR) services and can achieve high transmission rates, low latency, and reliability as has been shown in optical communications [7]. LDPC codes are capacity-achieving and are simpler to implement in practical systems compared to other channel coding. Further, they can support a wide range of block lengths and various code rates [8].

Another problem that arises in vehicular networks is the complexity of the underlying optimization problems that need to be solved to optimize the communications settings, such as code rate and modulation. Because of their complexity, these problems cannot be solved using conventional methods such as dynamic programming and exhaustive search. For such problems, greedy techniques result in sub-optimal solutions [9]. These algorithms cannot find the optimal solution in time-variant and dynamic contexts. To overcome these issues, deep reinforcement learning (DRL) has been used successfully in autonomous vehicular networks [10]. DRL employs deep Q-Network (DQN), a combination of Q-learning and deep neural networks, to approximate the state-action Q-value function by adjusting the weights of a neural network.

Several recent studies suggest the use of reinforcement learning (RL) in hybrid RF and Photodiode (PD)-based optical networks [11], [12]. In [11], the authors used RL for network selection, taking into account the traffic type and the possibility of having learning records to improve the Q-learning algorithm. The authors of [12] proposed an RL-based energy-efficient resource management scheme to improve energy efficiency. The systems described above employ traditional Q-learning, which is unsuitable for high-dimensional problems due to their slow convergence. Furthermore, they investigate a PD-based receiver, which encounters interference issues when dealing with multiple vehicles. OCC overcomes interference issues because it can spatially separate and process different transmitter sources independently on its image plane [4], allowing it to handle multiple users. More importantly, previous methods did not consider reliability and latency

Amirul Islam is with the Department of Electrical and Electronics Engineering, American International University-Bangladesh (AIUB), Bangladesh (e-mail: amirul@aiub.edu).

Nikolaos Thomos and Leila Musavian are with the School of Computer Science and Electronic Engineering, University of Essex, UK (e-mail: nthomos@essex.ac.uk, and leila.musavian@essex.ac.uk).

This research was funded in part by the Engineering and Physical Science Research Council, UK [grant number: EP/X012204/1]. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

constraints concurrently, making it impossible to guarantee that information is received reliably in the shortest amount of time.

In practice, vehicular network problems involve continuous variables, e.g., speed and distance, which we consider in this paper. To solve the continuous problem, most DRL algorithms discretize the action and state space, resulting in sub-optimal solutions that are far from the true solution [10], [13]. Discretization can affect the quality of the solution. If the discretization is too coarse, it may result in an inefficient sub-optimal solution; if it is too fine, it will take an enormous amount of time to find a solution with no guarantee of optimality. The above limitations can be addressed using actor-critic-based DRL frameworks [14]. To accelerate convergence, the Wolpertinger architecture [13] is used, in which the actions of the nearest neighbour are considered rather than exploring over a large actions space chosen by the actor network.

In our previous study [15], we investigated a Lagrange relaxation-based multi-agent deep RL vehicular OCC system, This multi-agent system leverages vehicles acting as autonomous agents to share information and make decentralized decisions based on local and received information from other vehicles, treating other vehicles as part of the environment. Although, the scheme in [15], improved spectral efficiency and decreased experience latency, it could not guarantee ultra-reliable services since it did not employ a channel coding scheme. Additionally, the discretization approach used in [15] is suboptimal, as we discussed earlier. To address these limitations, we proposed a single-link system in [16] using 5G NR LDPC codes and an actor-critic-based DRL framework to achieve uRLLC while maximizing throughput. However, this system does not consider information from other vehicular links. To address these limitations, in this paper, we present a throughput maximization scheme for multi-link vehicular OCC systems that optimizes the parameters (speed, code rate, and modulation scheme) of all available links to satisfy uRLLC requirements. We extend the work presented in [16] by expanding the problem definition and Markov decision process (MDP) formulation for multiple vehicular links. We employ an actor-critic-based DRL framework with the Wolpertinger architecture to deal with continuous and large state-action spaces. While actor-critic-based DRL has been used in various applications, the use of OCC for V2V communication in ITSs is a new and emerging area of research.

In summary, the contributions of this paper are highlighted as:

- We formulate a novel DRL-based throughput maximization scheme that optimizes LDPC code rates, modulation and adjusts vehicles' speeds while guaranteeing bit error rate (BER) and latency requirements, where we consider multiple links simultaneously.
- We use actor-critic-based Wolpertinger framework to address the problem of the continuous state-action problem and to avoid exploring large action spaces across all decision intervals.
- We thoroughly examine our algorithms' system performance, BER, and latency and compare them with other schemes under consideration. We also look at the convergence performance. The results show that the proposed method outperforms its counterpart schemes significantly. The results highlight the advantages of jointly optimizing the code rate, modulation and vehicles' speed and the consideration of the states from multiple vehicles to achieve uRLLC in the vehicular OCC system.

II. Related Works

Simultaneously achieving ultra-reliability and low latency in a dynamic environment, such as a vehicular network with significantly time-varying channels due to mobility and environmental factors, is a challenging task. Previous attempts to support uRLLC in cellular networks include a Markov chain-based link adaptation method proposed in [2], which maximizes link throughput while providing strict latency and block error rate requirements, and a network slicing solution for throughput maximization proposed in [3] for autonomous vehicular networks. However, the approaches proposed in [2] and [3] rely on RF technology, and hence these schemes suffer from interference. Moreover, these frameworks require centralized communication. Specifically, they require data to pass through a server for processing before being forwarded to the end users. This entails high latency and higher failure rates, potentially leading to loss of topological information of the traffic network due to delayed updates. In contrast, our DRL-based approach is decentralized, allowing each vehicle to make real-time decisions based on direct communication between vehicles without needing server-based processing. This helps maintaining up-to-date topological information, improving the communication rates and decreasing the perceived latency.

Various strategies, e.g., machine learning-based approaches [17] and frequency planning methods [18], are introduced to mitigate the interference in RF systems. These methods have high computational complexity and do not consider uRLLC requirements while they focus only on dealing with interference. Different from RF-based methods, OCC systems can easily deal with interference and other light sources, e.g., street lights or Sun, etc., while focusing on the specific pixels where the LEDs strike [4], [19]. Existing works using OCC mainly target to increase the data rate and ignore the uRLLC aspects that we study here. Specifically, in [4], the authors achieved 10 Megabits per second (Mbps) data rate by varying LEDs intensity while generating flag images from the communication image pixels in which the high-intensity light sources appear. In [19], the authors achieved a data rate of 20 Mbps per pixel without detection of LEDs and a data rate of 15 Mbps per pixel with real-time detection of LEDs in OCC systems. In [20], the achievable transmission rate was further improved to 54 Mbps for BER $< 10^{-5}$ at a 50 m distance.

Vehicular networks are dynamic and time-varying, often involving a large number of vehicles and variables that increase the complexity of the problem. This prohibits the use of stochastic optimization methods in an online manner. The complexity of the problems faced in vehicular networks can make it difficult to solve them fast, which can lead to

TABLE I
COMPARISON OF EXISTING STUDIES AND OURS

Ref	Communication Technology	Optimization			Ultra Reliable	Low Latency Requirements
		Centralized/ Distributed	Machine learning enabled or traditional	Single Link/ Multiple Links		
[2]	RF	Centralized	Traditional optimization	Single Link	No	Yes
[3]	RF	Centralized	Traditional optimization	Single Link	No	Yes
[4]	Vehicular OCC	Decentralized	No Optimization	Single Link	No	No
[19]	Vehicular OCC	Decentralized	No Optimization	Single Link	No	No
[20]	Vehicular OCC	Decentralized	No Optimization	Single Link	No	No
[10]	RF	Centralized	Machine learning enabled	Multiple Links	No	No
[21]	RF	Decentralized but using edge server for processing	Machine learning enabled	Multiple Links	No	Yes
[22]	RF	Decentralized but with Roadside unit	Machine learning enabled	Multiple Links	No	No
[23]	RF	Centralized	Machine learning enabled	Multiple Links	Reliable	Yes
[15]	Vehicular OCC	Decentralized	Machine learning enabled	Multiple Links	Reliable	Yes
[16]	Vehicular OCC	Decentralized	Machine learning enabled	Single Link	Yes	Yes
Our	Vehicular OCC	Decentralized	Machine learning enabled	Multiple Links	Yes	Yes

Note: low latency: < 10 ms, reliable: $> 10^{-5}$, ulltra-reliable: $< 10^{-7}$

violations of the low latency constraint. As a result, it is challenging to meet the requirements of uRLLC in vehicular networks. DRL has emerged as an optimization framework to deal with the time-varying nature and the complexity of the optimization problems in vehicular networks [10]. However, decision-making parameters, e.g., speed, and distance, are continuous and therefore, general DQN cannot be trivially applied without discretization of the action-state spaces, which leads to performance degradation [14]. Several recent studies in RF systems exist, but they do not simultaneously consider both decentralized approaches and uRLLC. For instance, the work in [21] proposes a decentralized resource allocation scheme for vehicle-to-infrastructure communication that minimizes interference but relies on an edge server for information processing and decision-making. Similarly, in [22], the authors utilize a multi-agent DQN architecture for decentralized communication to improve resource utilization and latency compared to a centralized approach, but it fails to meet uRLLC requirements. The authors in [23] present a centralized RF-based power allocation method using a cooperative DRL scheme to achieve uRLLC. Motivated by these limitations mentioned throughout this section, in this paper, we propose a DRL-based scheme for multiple vehicular links in order to cope with the diverse nature of the vehicular OCC systems, while we follow an actor-critic-based DRL framework with Wolpertinger policy architecture. Table I provides a summary comparing existing related studies with our work in terms of communication technologies, optimization schemes, and uRLLC satisfaction.

III. SYSTEM MODEL

A. System Overview

Our proposed solution considers the vehicular OCC setting shown in Fig. 1, where each vehicle has a transmitting unit in the back with LED backlights and a vision camera set, as well as a receiving unit in the front with a high-speed camera (1000 frames per second (fps)). The front camera serves two purposes. It can measure the distance between

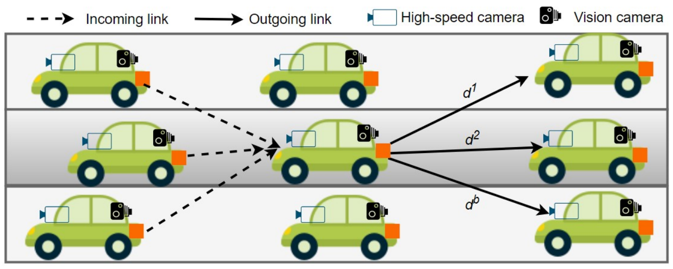

Fig. 1. Proposed multi-agent vehicular OCC system model.

the agent vehicle and the vehicle in front, and it is also the receiver that decodes data transmitted from the front vehicle's LED transmitters. The back camera uses a stereo-vision camera, similar to the one described in the study [24], to measure the distance behind the vehicle. As shown in Fig. 1, each vehicle receives information from the front vehicle and transmits information to the backward vehicles. The information includes the vehicle's moving intentions (for example, braking, accelerating, changing lanes), emergency information, and so on. We denote the number of vehicle-to-vehicle (V2V) links at the back of each vehicle as B and $\mathcal{B} = \{1, 2, \cdots B\}$ represents the set of V2V links. The distance with the backward vehicles is expressed as d^b, where $b \in \mathcal{B}$, and b represents the index of the backward V2V link.

In our system, we employ an adaptive M-ary quadrature amplitude modulation (M-QAM) scheme as it offers low BER and improved spectral efficiency [5]. Other modulation techniques can also be used. Additionally, we utilize time division multiple access (TDMA) to transmit at various modulation orders for different vehicles behind each vehicle. In TDMA, each V2V link transmits at a specific time instant only. This is done by assigning specific time slots for each link during transmission or reception, thus the spectral efficiency is computed by dividing it by the number of users (vehicles), B, that are located behind the vehicle. To improve the transmission rate and ensure uRLLC, we employ 5G NR LDPC codes in our system. The overall system block diagram is illustrated in

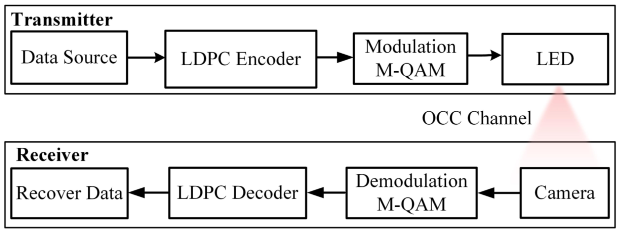

Fig. 2. Block diagram of LDPC coding with M-QAM for vehicular OCC system.

Fig. 2, which includes a transmitter, a channel and a receiver. At the transmitter, the data bits are encoded using LDPC codes and then are mapped into M-QAM symbols by adaptive modulation. Following that, the coded data is transmitted via LEDs over the OCC channel. The receiver uses the camera to capture the modulated light intensity and then uses them to recover the originally transmitted data using the M-QAM [25]. Finally, an LDPC decoder decodes the demodulated codeword (or message). More information on the M-QAM encoding and decoding process can be found in [15]. The acronyms used in this study are summarized in Table II.

B. Channel Modelling

We consider uninterrupted line-of-sight (LoS) communication links between the transmitter and receiver vehicles to support continuous communication. In our proposed OCC system, we adopt the channel model from [15].

As stated in [15], the achievable transmission rate of the OCC system for the link b using the M-QAM modulation scheme and 5G NR LDPC codes with code rate, $\varkappa$, is given by:

$$C^b(\varkappa) = \frac{W_{\text{fps}}}{3} \cdot \frac{\varkappa\, N_{\text{LEDs}} w \varrho}{2\, \tan\left(\frac{\theta_l}{2}\right) \cdot d^b} \cdot \log_2(M^b), \tag{1}$$

where W_{fps} stands for the camera's sampling rate, N_{LEDs} denotes the number of LEDs in each row of the transmitter, w represents the image width, ϱ is the size of LED lights in cm^2, and M^b is the available constellation points for each V2V link b, e.g., $M = 4, 8, 16, \cdots$. Please note that, the distance d^b in (1) is affected by the relative speed of the vehicle v, which affects the vehicle's position on the road. The inter-vehicular distance at current time t is adjusted by $d_t = d_{t-1} + v_t \cdot \Delta t$, where d_{t-1} is the distance of the previous time instance and Δt is the time elapsed between time instant t and $t-1$.

The transmission latency dominates the end-to-end latency because we process a small amount of data, i.e., the decision information from transmitter to receiver. The transmission latency for packet size, L, is therefore stated as $\tau^b(\varkappa) = L/C^b(\varkappa)$, in accordance with [15].

IV. Problem Formulation and Proposed Solution

A. Constrained Problem Formulation

Here, we formulate an optimization framework to maximize the communication rate of the proposed vehicular environment while meeting uRLLC requirements. Specifically, we formulate an optimization problem that maximizes the sum throughput of the vehicular OCC system by selecting the

TABLE II
List of Acronyms

Acronym	Description
5G	Fifth Generation
AV	Autonomous Vehicle
BER	Bit Error Rate
DDPG	Deep Deterministic Policy Gradient
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
fps	frame per second
ITS	Intelligent Transportation System
KNN	K-Nearest Neighbour
LDPC	Low-Density Parity-Check
LED	Light-Emitting Diode
LoS	Line-of-Sight
Mbps	Megabits Per Second
MDP	Markov Decision Process
MIP	Mixed-Integer Programming
mph	miles per hour
M-QAM	M-ary Quadrature Amplitude Modulation
NN	Neural Network
NR	New Radio
OCC	Optical Camera Communication
PD	Photodiode
ReLU	Rectified Linear Unit
RF	Radio Frequency
RL	Reinforcement Learning
RMSPro	Root Mean Square Propagation
SLO	Single Link Optimization
SNR	Signal-to-Noise Ratio
SUMO	Simulation of Urban Mobility
TDMA	Time Division Multiple Access
TraCI	Traffic Control Interface
uRLLC	Ultra-Reliable and Low-Latency Communication
V2V	Vehicle-to-Vehicle

optimal values for the modulation order, the code rate and the relative speed of the vehicle. We set the BER and latency to a predefined value to respect the uRLLC conditions imposed by the system. Finally, we define the constrained problem as:

$$\max_{\mathcal{M},\mathcal{X},v} \quad \frac{1}{B}\sum_{b=1}^{B} C^b(\varkappa), \tag{2}$$

$$\text{s.t.} \quad \text{BER}^b(\varkappa) \leq \text{BER}_{\text{tgt}},\ \forall b; \tag{3}$$

$$\tau^b(\varkappa) \leq \tau_{\max},\ \forall b; \tag{4}$$

$$M^b \in \mathcal{M},\ \forall b; \tag{5}$$

$$\varkappa^b \in \mathcal{X}.\ \forall b; \tag{6}$$

where $\mathcal{M}$ is the set of QAM modulation orders, $\mathcal{X}$ represents the LDPC codes set, BER_{tgt} denotes the maximum target BER, and the maximum allowable latency is defined by $\tau_{\max}$. To guarantee reliability the target BER must be satisfied as shown in (3) and the low latency condition should be met by maintaining $\tau_{\max}$ as in (4). We choose the modulation scheme from $\mathcal{M}$, which is given in (5). The code rates are adapted from the available 5G NR LDPC codes [8], which is defined in the IEEE standard and is shown in (6).

To meet the requirements of uRLLC in our vehicular OCC system, we set threshold values for delay and reliability based on the requirements of the specific use case. The specific requirements for uRLLC can vary based on the application, but for vehicular communication, a packet error rate of 10^{-5} [26] and a latency of 3 to 10 ms for a packet of 300 bytes is typically considered sufficient [27]. For larger packet sizes, such as 5 kbits in our case, a maximum latency of 10 ms would meet the requirements for vehicular communication [27]. In terms of reliability, we have set the maximum BER at 10^{-7}, which is measured by communicating a 5 kbits packet.

The studied problem in (2) is mixed-integer programming (MIP) with nonlinear constraints for BER in (3) and delay in (4). As a result, our problem is non-deterministic polynomial-time-hard [28]. MIP problems are known to have high computational complexity [29]. The problem studied in this paper cannot be solved using traditional methods such as dynamic programming and exhaustive search because they are computationally complex and slow to convergence. The decision space for our problem is large due to the many possible values for our control variables (speed, code rate, and modulation). To address these challenges, we have chosen to use deep RL which allows us to solve the problem with less computational and time complexities.

B. Modelling of MDP

The proposed multi-agent maximization problem in (2) - (6) is formulated as an MDP, which is outlined as a tuple ($\mathcal{S}$, $\mathcal{A}$, p, r, ζ) [30], where $\mathcal{S}$ is the set of all possible states; $\mathcal{A}$ denotes the set of all possible actions; $p(s_{t+1}, r_t|s_t, a_t)$ denotes the transition probability which describes the probability that an agent selects an action $a_t \in \mathcal{A}$ and transits to a new state $s_{t+1} \in \mathcal{S}$ from the current state $s_t \in \mathcal{S}$; while r represents the reward. The discount factor is represented by the parameter $\zeta \in [0, 1]$, which gradually discounts the effect of actions on future rewards. We outline the state space $\mathcal{S}$, the action space $\mathcal{A}$, and reward r of the considered RL framework as follows:

*1) **State space:*** At time t, each agent interacts with the environment and observes the state, s_t. The state in our system has three parts: the backward distance vector, $\mathbf{d}_t^b = (d_t^1, \cdots, d_t^B)$, the transmitted modulation scheme, $\mathbf{M}_t^b = (M_t^1, \cdots, M_t^B)$, from the set $\mathcal{M}$ = {4, 8, 16, 32, 64}, and the code rate vector, $\boldsymbol{\varkappa}_t^b = (\varkappa_t^1, \cdots, \varkappa_t^B)$, from the set $\mathcal{X}$. In summary, the state is outlined as $s_t = \left\{\mathbf{d}_t^b, \mathbf{M}_t^b, \boldsymbol{\varkappa}_t^b\right\}$

*2) **Action space:*** From the state s_t, the agent takes an action a_t from the set $\mathcal{A}$, consisting of adjusting the relative speed, v_t, selecting modulation scheme $\mathbf{M}_t^b \in \mathcal{M}$, and code rate $\boldsymbol{\varkappa}_t^b \in \mathcal{X}$. We summarize the action space as $a_t = \left\{v_t, \mathbf{M}_t^b, \boldsymbol{\varkappa}_t^b\right\}$.

*3) **Reward function:*** The agent receives a reward based on the action, a_t, taken from the state, s_t. The reward function in our framework is a weighted sum of rewards corresponding to inter-vehicular distance, BER constraint (3), latency constraint (4), and throughput (2). We first model the reward for the distance changes, r_t^{d}, as follows:

$$r_t^{\text{d},i} = \begin{cases} -1 \times (d_{\text{stop}} - d_t^b), & d_t^b < d_{\text{stop}} \ , \\ \frac{1}{d_t^b - d_{\text{stop}}}, & d_t^b > d_{\text{stop}} \ , \end{cases} \tag{7}$$

where i is the index of the agent. Please recall that, d_t^b is the backward distance of the vehicle, but in designing our reward, we only consider the distance with the vehicle behind residing in the same lane on the road as the primary goal is to avoid collision with the vehicle in the same lane. This is the decisive vehicle as it has the potential to approach the agent vehicle in the next time step or near future. d_{stop} denotes the stopping distance, which is equal to the sum of distance travelled by the vehicle after the brakes are applied and the distance travelled due to the driver's reaction time after observing a situation [31]. From hereon, we drop i for notational simplicity. To satisfy the BER and latency requirements, we model the reward for BER, r_t^r, and latency, r_t^τ, as:

$$r_t^r(\varkappa) = \mathbb{1}_{\text{r}}(\text{BER}_t^b(\varkappa) \leq \text{BER}_{\text{max}}), \tag{8}$$

$$r_t^\tau(\varkappa) = \mathbb{1}_\tau(\tau_t^b(\varkappa) \leq \tau_{\text{max}}), \tag{9}$$

where $\mathbb{1}_{\text{r}}$ and $\mathbb{1}_\tau$ stand for the indicator functions of the BER and latency, respectively.

Considering the above definition, we can express the overall weighted sum of the rewards, r_t, as

$$r_t = \omega_d \ r_t^{\text{d}} + \omega_r r_t^r(\varkappa) + \omega_\tau r_t^\tau(\varkappa) + \frac{\omega_c}{B} \sum_{b=1}^{B} C^b(\varkappa), \tag{10}$$

where, ω_d, ω_r, ω_τ, and ω_c are positive weights that are used to balance the distance, BER, latency, and sum throughput rewards. The reward (10) is designed such that the violation of reliability and delay constraints (Equations (8) and (9), respectively) result in zero reward. This design guides the learning process towards solutions that consistently meet the uRLLC requirements.

After each interaction with the environment in time slot, t, the agent receives a reward r_t. The goal of RL is to maximize the total future discounted reward: $G_t = \sum_{j=0}^{\infty} \zeta^j r_{t+j+1}$.

C. Proposed Solution

Q-learning is a well-known approach for solving MDP problems [30]. However, the size of the state-action set affects how quickly Q-learning converges. The algorithm converges fast for small state-action spaces since the agent can quickly explore the state-action pairs and determine the optimal policy. On the contrary, as the Q-table grows bigger for large state-action spaces, the convergence rate slows down. Additionally, the complexity of computation grows linearly with the size of the state-action set for multi-agent systems. Hence, for large state spaces, the problem becomes intractable, necessitating a longer time to converge and a large amount of memory to store the Q-table. Moreover, there is no guarantee that the derived solution will be optimal. This happens as a large number of state-action pairs may not be visited at all.

Since the vehicular environment is dynamic and time-variant, we have continuous space because of the nature of our variables, e.g., speed and distance. The discretization of the state-action space is a popular method for dealing with the continuous problem. However, there is a trade-off between performance and state-action space size. During discretization, the state-action space parameters are generalized by sacrificing the performance. Furthermore, Q-learning cannot directly be applied to continuous problems as it does not employ an approximation function (neural network) and the only approximation comes by quantizing the state-action space. When we have a large problem, the optimization becomes practically too slow. To overcome the issues raised above, we propose an actor-critic framework based on the deep deterministic policy gradient (DDPG) algorithm [32]. To stabilize the training process, we utilize a new policy architecture called Wolpertinger architecture [13]. This architecture avoids the extensive computational cost of evaluating the Q-function on every action taken and instead, it evaluates the actions of the nearest neighbour. We have used the Actor-Critic based DRL approach for our sequential decision problem, where the actions of one vehicle affect the actions in the next time slots. This approach has been shown to be effective in solving sequential decision problems as it learns to make decisions by maximizing a reward function, which can lead to better long-term performance compared to traditional optimization methods [32].

D. Wolpertinger Architecture

The proposed multi-agent DRL algorithm is based on an actor-critic framework that follows the Wolpertinger architecture [13]. The Wolpertinger framework is particularly attractive due to its ability to handle large action spaces efficiently through a combination of a K-nearest neighbours (KNN) algorithm and actor-critic framework. This method ensures reduced computational complexity and enhanced long-term outcomes by optimizing rewards, even in high-dimensional environments, where traditional frameworks often struggle with increased computational complexity and slower convergence times.

This policy architecture has three major elements: actor network, KNN, and critic network, which work in three stages. Firstly, the actor takes states as the input and constructs a proto-actor, $\hat{a}$, as output. Secondly, the proto-actor is fed as the input to the KNN, which computes the L_2 distance between the proto-actor and each valid action and keeps a list of the K actions that result in the smallest L_2 distance. In this way, the proto-actor is expanded over the action space, $\mathcal{A}_K$, where K is the number of elements and every element is an action $a \in \mathcal{A}$. Finally, the critic network takes $\mathcal{A}_K$ as input while refining the actions of the actor network based on the maximum Q value. In order to update the actor and critic networks, we train the policy using the DDPG algorithm [14]. We outline the fundamental components of the actor-critic policy architecture below.

*1) **The actor network:*** The actor network maps the state $s \in \mathcal{S}$ to the corresponding action space and chooses a proto-actor $\hat{a} \in \mathcal{A}$ from the valid actions. The network is characterized as θ^{μ}. Thus, the proto-actor is defined as: $\mu(s \mid \theta^{\mu}) : \mathcal{S} \to \mathcal{A}$ and $\mu(s \mid \theta^{\mu}) = \hat{a}$.

*2) **K-nearest neighbours (KNN):*** For a large action space, the generation of proto-actors reduces computational complexity. However, using a single actor to represent the entire action space can lead to suboptimal decisions. To address this issue, KNN mapping, g_K, is used by expanding the actor's choice of action to valid action subsets from $\mathcal{A}$. We express the returned action set, $\mathcal{A}_K$, from g_K as $\mathcal{A}_K = g_K(\hat{a}_t)$, with

$$g_K = \arg\overset{K}{\min_{a \in \mathcal{A}}} \mid a - \hat{a} \mid^2, \tag{11}$$

where $\mid a - \hat{a} \mid^2$ denotes the features distance between the proto-actor $\hat{a}$ and the selected action a. After selecting the proto-actor by the actor network, the agent determines the KNN feature distances by exploring the action space and accordingly, the action set can be formed. We can find the K nearest neighbours using (11).

*3) **The critic network:*** To avoid selecting actions with low Q-values that lead to poor decisions, the critic network is introduced, which refines the actions chosen by the actor. The deterministic policy in the critic network is characterized as:

$$Q\left(s_t, a_t \mid \theta^Q\right) = \mathbb{E}\left[r(s_t, a_t) + \zeta Q\left(s_{t+1}, a_{t+1} \mid \theta^Q\right)\right], \tag{12}$$

where θ^Q is the parameter of the critic network. The critic calculates the Q value while considering the current state, s_t, and the next state, s_{t+1}, as the input. The critic network evaluates all actions in $\mathcal{A}_K$ and chooses the action which gives the maximum Q-value, as follows:

$$a_t = \arg\max_{a_t \in \mathcal{A}_K} Q(s_t, a_t \mid \theta^Q). \tag{13}$$

Update: At each timestep, a minibatch is sampled uniformly from the replay memory to update the actor and critic networks. Since DDPG is an off-policy algorithm, it allows the algorithm to benefit from learning across a set of uncorrelated transitions. Hence, we update the actor policy using DDPG with a minibatch size $N_{\mathcal{B}}$, which is expressed as

$$\begin{aligned}\nabla_{\theta^{\mu}} J \approx & \frac{1}{N_{\mathcal{B}}} \sum_t \nabla_a Q\left(s, a \mid \mu^Q\right) \mid_{s=s_t, a=\mu(s_t)} \\ & \nabla_{\theta_{\mu}} \mu\left(s \mid \theta^{\mu}\right) \mid s_t,\end{aligned} \tag{14}$$

and the critic is updated by minimizing the loss:

$$L = \frac{1}{N_{\mathcal{B}}} \sum_t \left(y_t - Q\left(s_t, a_t \mid \theta^Q\right)\right)^2, \tag{15}$$

where the target network is derived as

$$y_t = r_t + \zeta Q'\left(s_{t+1}, \mu'(s_{t+1} \mid \theta^{\mu'}) \mid \theta^{Q'}\right). \tag{16}$$

However, evaluating the loss in (15) using deep neural networks is difficult. This is due to the fact that the target value of (16) is determined using the updated $Q(s, a \mid \theta^{\mu})$ value, which suffers from divergence issue when attempting to find the optimal solution. To solve the above problem, we employ a target network similar to [33]. This is done using "soft" target updates for actor-critic networks. In this way, we calculate the target values by copying the actor and critic networks, $\mu'(s \mid \theta^{\mu'})$ and $Q'(s, a \mid \theta^{\mu'})$, respectively. We then update the target network's weights by slowly tracking the networks using

$$\theta^{Q'} \leftarrow \beta\theta^Q + (1 - \beta)\theta^{Q'}, \tag{17}$$

$$\theta^{\mu'} \leftarrow \beta\theta^{\mu} + (1 - \beta)\theta^{\mu'}, \tag{18}$$

where $\beta \ll 1$ denotes the soft target update rate. This helps to improve the learning performance while changing the target values slowly. DDPG help us decoupling the exploration problem from the learning algorithm. Hence, we define the exploration policy μ' by sampling the noise, n_t, from the noise

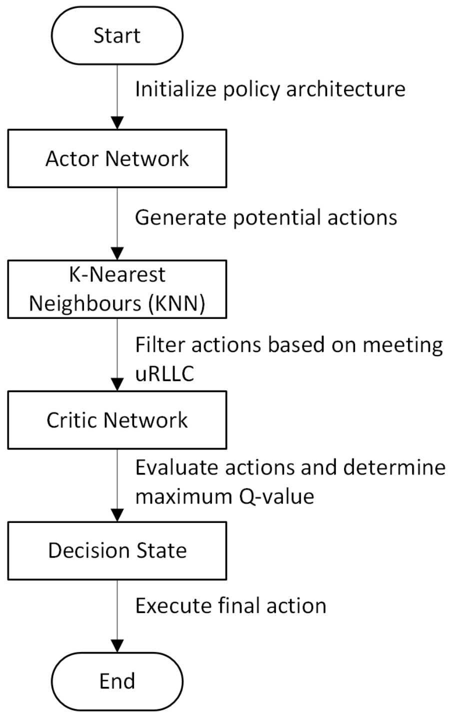
Start
Initialize policy architecture
Actor Network
Generate potential actions
K-Nearest Neighbours (KNN)
Filter actions based on meeting uRLLC
Critic Network
Evaluate actions and determine maximum Q-value
Decision State
Execute final action
End

Fig. 3. Architecture of Actor-critic based DRL scheme for Vehicular uRLLC.

process and adding it to the actor policy $\mu'(s_t) = \mu(s_t \mid \theta_t^{\mu}) + n_t$, where n_t is selected to fit the environment. We assume temporally correlated noise to explore the environment satisfactorily using a similar process to that presented in [34]. While there is no general theoretical guarantee that the optimal action is among the nearest neighbours of a proto-actor, several studies have shown the effectiveness of the KNN algorithm in similar problems [35], [36]. The effectiveness of our approach is demonstrated through experiments conducted in the performance evaluation section.

The policy architecture for attaining uRLLC by combining the actor network, KNN, and the critic network—is illustrated in the state diagram in Fig. 3. The policy architecture is initialized in the Start stage. After that, the system transitions to the Actor Network stage, which is responsible for generating potential actions (speed, modulation schemes, and code rate) based on the current state of the environment. Then, these actions are passed to the KNN state, where KNN mapping is utilized. This involves expanding the actor's choice of actions to valid action subsets and determining the KNN feature distances. By exploring the action space, KNN selects the most prevalent actions that meet the uRLLC conditions. These actions are then evaluated in the Critic Network stage, which decides the maximum Q-value for each action, guiding the optimal action selection. The selected action is then evaluated in the Decision stage, where it is executed. The procedure concludes in the End stage, completing the decision cycle. Each agent determines the derived policies independently.

E. Complexity Analysis

The Wolpertinger algorithm's training process has a time complexity dependent on the amount of training data and duration, but we focus on the running process. The running time complexity is characterised by the neural network structure and the state-action space dimension. Unlike traditional stochastic non-convex methods, the computational time is linked to the critic network's convergence rate, which impacts the overall performance. Slow critic convergence bottlenecks actor-critic methods, while fast convergence shifts the challenge to policy gradient updates. Experimental results show that faster critic convergence leads to quicker actor-critic convergence, though often reaching suboptimal stationary points, highlighting the interplay between optimization, generalization, and function parameterization in RL.

The Wolpertinger architecture's time complexity can be outlined as follows:

- Actor Network: The time complexity of the actor network, which involves processing the input state and generating an action, is derived and expressed as $O(\|A\|)$, where $\|A\|$ represents the number of actions considered.
- K-Nearest Neighbors (KNN): For the KNN component, the time complexity is $O(N \log N)$ for sorting the states and $O(K)$ for the nearest neighbour search, where N is the number of observations and K is the number of nearest neighbours.
- Critic Network: The critic network evaluates the chosen actions, and its time complexity is derived as $O(\mathtt{C})$, where $\mathtt{C}$ represents the number of evaluations performed.

Considering the interactions between these components, the overall time complexity of the framework can be approximated as $O(A + N \log N + K + \mathtt{C})$.

In practice, however, increasing the K value above a certain point does not result in improved performance. The authors of [13] show that increasing the value of K leads to a significant improvement in performance, even though it may render other areas of performance, for example, higher convergence rate and computational time. When only 5% or 10% of the maximum number of actions are used, the method performs similarly to when the entire action set is used. Using the remaining actions would result in relatively minor performance gains while significantly increasing computational time. Therefore, in our case, we use 7% of the available action set for the KNN algorithm.

We would like to note that while actor-critic DRL methods have shown significant empirical success, they lack strong theoretical guarantees of global optimality. Actor-critic algorithms typically converge to local optima under certain conditions, such as when using neural network function approximation [37]. However, due to their gradient-based nature, global optimality is not assured [30]. Although theoretical bounds exist for frameworks like Multi-armed Bandits, cannot be used with our problem due to the dimensionality of the studied problem. Our proposed scheme is generic, without assumptions limiting its applicability to special scenarios. Unlike supervised and semi-supervised methods, our approach can be applied both offline when there is available data to train the DQN model and online to update the DRL model. Our DRL model can run only online, i.e., without pretraining, but with the tradeoff of slower convergence.

V. Evaluation

A. Simulation Settings

1) SUMO Framework: We implement our proposed vehicular environment in Simulation of Urban Mobility (SUMO) framework [38]. SUMO is a multi-model traffic and extendable, open-source, microscopic, and widely used simulator. It offers a wide range of monitored quantities for various traffic simulation scenarios, including vehicle routing, traffic light control, and multi-modal transport simulations. Its compatibility with other tools and frameworks ensures seamless integration into our research workflow, enhancing the robustness and functionality of our simulations. Additionally, it enables users to build specific traffic scenarios on given road maps, ensuring scalability and adaptability. It also supports traffic control interface (TraCI), a Python-based application programming interface to adapt the simulation online. In SUMO, various sets of driver models already exist, and it is relatively simple to add more models. We adjusted the SUMO environment according to the requirements of our proposed multi-agent vehicular system by changing some settings. For example, the simulation window size is 180m and the maximum number of vehicles per timestep on the window is 20. We introduce diversity in our system by including a few aggressively moving vehicles to the SUMO environment. The simulation parameters for the SUMO framework are listed in Table III.

After the training is initiated, the vehicles are loaded in the SUMO following the specified settings in Table III. The interaction between the SUMO framework and the DRL agent is controlled by the TraCI interface. Using TraCI, the DRL agent receives different information about the vehicular network, including distance, speed, position of the vehicle, and applies the action on the environment following the policy found by the DRL agent which applies our framework presented in Section IV. The decision is fed back to the SUMO again. Thus, the agent updates its state following taking the action while moving to the next state in the SUMO environment. Then, the reward is computed as in (10) and communicated to the agent at every simulation run. This process continues until the maximum number of iterations is reached or a convergence threshold is achieved. We note that, in our simulation, we consider timestep as the decision interval.

To generate our dataset we used SUMO. Specifically, we collected various measurements, including inter-vehicular distance, relative speed of the vehicles, and their positions within the simulated environment, from the SUMO environment through the TraCI interface. The TraCI interface acted as the communication bridge between SUMO and the DRL agent, feeding this essential data to the agent which is used to understand traffic dynamics and make informed decisions.

2) Training Parameters: For OCC system design, we consider the communication of 10^{11} bits and a packet size of 5 kbits, where we train the model with the transmission of zero codewords, i.e., all the bits of the codeword are zero, which are sufficient for the training as the channel is symmetric [39]. We use the code rates of the 5G NR LDPC codes as defined in the IEEE standard [40]. In the simulation, an actor-critic-based DRL framework is employed with the following training parameters and settings. The actor and critic networks have four fully connected layers: an input layer, two hidden layers with 500 and 250 neurons, and an output layer. The state space incorporates the distance, modulation scheme, and code rate, and hence, the input layer has $(N_d + |\mathcal{M}| + |\mathcal{X}|)$ nodes, where $N_d = 150$; $|\mathcal{M}| = 5$; and $|\mathcal{X}| = 20$; with N_d being the number of neural nodes for distance. We consider a distance of up to 150 meters, which is sufficient for maintaining good communication quality and avoiding collisions, and $|\mathcal{M}| = 5$ as only 5 modulation schemes are used. While the output layer, as in our proposed system, has $(\triangle M + \triangle \varkappa + \triangle v)$ nodes, the action involves changes in modulation scheme, code rate, and velocity where $\triangle M = 5$, $\triangle \varkappa = 20$, and $\triangle v = 60$. We utilize rectified linear unit (ReLU) activation function [41] for all the layers. We use TensorFlow in our evaluations to implement deep reinforcement learning algorithms. To minimize

TABLE III
SUMO MODELLING PARAMETERS

Parameter	Value
Initial velocity of vehicle	5 mph
Window size of the simulation	180 m
Maximum number of vehicle per window	20
Number of lane	3
Step length	1 m
Lateral movement of vehicle	0.64 m per timestep

Algorithm 1 Actor-Critic Algorithm

```
1: Initialize the actor $\mu(s \mid \theta^\mu)$ and critic $Q\left(s, a \mid \theta^Q\right)$ networks randomly with weights $\theta^\mu$ and $\theta^Q$.
2: Initialize the target networks $\mu'$ and $Q'$, while also updating the weights by $\theta^{\mu'} \leftarrow \theta^\mu$, $\theta^{Q'} \leftarrow \theta^Q$
3: Initialize SUMO environment parameters and replay memory according to system requirements.
4: for each episode do
5:     Observe the initial state $s_t$
6:     for each timestep $t$ do
7:         Observe current state $s_t$
8:         Actor: Receive proto-action from actor network $\hat{a}_t = \mu(s_t \mid \theta^\mu)$.
9:         KNN: Locate the approximated $k$ nearest actions $\mathcal{A}_K = g_K(\hat{a}_t)$
10:        Critic: Choose action $a_t = \arg\max_{a_t \in \mathcal{A}_K} Q(s_t, a_t \mid \theta^Q)$ in accordance with the current policy
11:        Apply action $a_t$ to the environment; and observe reward $r_t$ and new state $s_{t+1}$
12:        Store transition $(s_t, a_t, r_t, s_{t+1})$ in replay memory.
13:        Randomly sample a mini-batch of $N_\mathcal{B}$ transitions from the replay memory
14:        Set target $y_t = r_t + \zeta Q'\left(s_{t+1}, \mu'(s_{t+1} \mid \theta^{\mu'}) \mid \theta^{Q'}\right)$
15:        Update critic by reducing loss with (15)
16:        Use the sampled policy gradient to update the actor policy with (14)
17:        Update target networks with (17) and (18) having $\beta \ll 1$.
18:    end for
19: end for
```

the loss function and update DQN network parameters, we employ root mean square propagation (RMSProp) optimizer as the training algorithm. To make sure that the critic network learns more quickly than the actor network, we set the learning rate of the actor network to 0.0001 and the learning rate of the critic network to 0.001. The soft target value is set at $\beta = 0.001$, which is sufficient to balance between the optimality and computational cost.

We train the actor critic based DRL scheme for a total of 10000 timesteps, which is sufficient for convergence. To effectively explore the environments, we use temporally correlated noise in the exploration noise process. Specifically, we use the Ornstein-Uhlenbeck process models [34] with a mean value of 0.15 and a variance of 0.2, resulting in temporally correlated values centred around 0. We train the network with mini-batches of 64 samples and a replay buffer size of 10^{11} to save the transitions in memory. We set the discount factor ζ to 0.98. We also normalize different sub-rewards corresponding to distance, BER, latency, and transmission rate in (10) so that they are on a similar scale. Specifically, this normalization is done by scaling the reward function for distance (7) and rate of (2) to keep the scale of (10) between 0 and 1. This improves the neural network (NN) model's convergence speed and training stability. Table IV lists the simulation parameters.

We would like to point out that in our simulation, the vehicle's speed changes by intervals of 0.5 miles per hour (mph) which was selected for illustration purposes. In our framework, each vehicle changes its speed independently based on the forecast reward (which depends on the distance, and the conditions for reliable communication). Our reward formulation includes distance to maintain a safe distance between the vehicles as shown in (7), which penalizes the agent vehicle if the safety distance is violated. Hence, the agent is discouraged to keep the distance too small.

3) Training Algorithm: Algorithm 1 outlines the training procedure of the actor-critic based DRL algorithm we propose, which is executed on all vehicles within the simulation environment. An agent observes the state s_t (distance, modulation scheme, and code rate) on line 5 in each training step. The actor network then identifies a proto-actor, $\hat{a}_t$, following the policy on line 8, which is then expanded into an action set $\mathcal{A}_K$ using KNN mapping on line 9. The action set (change in modulation, coding rate, and velocity) is evaluated by the critic network to identify the action set that can deliver the highest state value on line 10. On line 11, these actions are then applied to the environment. After each timestep, the resultant reward and the following state are recorded in the replay buffer along with the performed action, which is (s_t, a_t, r_t, s_{t+1}) as shown on line 12. On line 13, a random transition is sampled from the replay buffer, and on line 14, the target Q value is updated using the target network's weights by applying the (16) function. The actor is then trained on line 16 by using (14) based on the policy gradient after the critic parameter has been adjusted on line 15 by minimising the loss using (15). The target network is then modified by gradually changing the weights of (17) and (18) on line 17. This enables the learning process the opportunity to use previously ignored information about which action was actually performed to train the critic while taking the policy gradient at the actual output.

TABLE IV
SIMULATION PARAMETERS

Parameter, Notation	Value
Camera-frame rate, W_{fps}	1000 fps
Number of LEDs at each row, N_{LEDs}	30
Packet size, L	5 kbits
Size of the LED, ϱ	15.5 $\times$ 5.5 cm^2
Resolution of image, w	512 $\times$ 512 pixels
Mini-batch size, $N_{\mathcal{B}}$	64
Replay memory size	10^{11}
Number of hidden layer (Neurons)	$2(500, 250)$
Discount factor, ζ	0.98
Exploration rate, ϵ	0.05
Learning rate (Actor network)	0.0001
Learning rate (Critic network)	0.001
Soft target updates rate, β	0.001
Gradient momentum (used by RMSProp)	0.95

While the training process can lead the model to achieve good performance eventually, it is important to acknowledge that reliable performance may not always be guaranteed before convergence. To address this practical challenge, we implemented a two-phase training approach. In the first phase, the model undergoes pre-training offline using historical data. This allows the model to establish a foundational understanding. Subsequently, the model is deployed online where it updates the policies using real-time data based on encountered situations. This approach ensures the model leverages both pre-existing knowledge and adapts to real-world conditions, avoiding the need to start training entirely from scratch during deployment.

B. Comparison Schemes

We evaluate the performance of the proposed actor-critic-based DRL scheme against different variants of the proposed scheme. Before presenting simulation results, we first provide an overview of the various schemes under comparison as:

1) ***Proposed scheme:*** We call our actor-critic vehicular OCC system as the proposed scheme. In our scheme, we use the configurations as we mentioned in Section V-A2. We set the discount factor to 0.98. This scheme is optimized by employing the code rate optimization and controlling speed and modulation described in Section V-A2, in which we observe all of the links behind the agent vehicle and optimize the policy based on these observations. Different modulation schemes and code rates are chosen for each vehicular link.

2) ***No Coding [15]:*** In this scheme, we consider a multi-link system similar to our proposed scheme presented in [15] that does not use channel coding. Hence, this scheme is referred to as the "No Coding" scheme. The purpose of comparing this scheme with others is to understand the impact of channel coding on system performance. In the No Coding scheme, we maximize the sum spectral efficiency without performing the code rate optimization while adopting the algorithm shown in Section V-A2. This scheme only considers the latency constraint, as the reliability constraints of uRLLC cannot be met without channel coding.

3) ***Single Link Optimization [16]:*** In this scheme, we only observe the state of the link with the vehicle behind in the same lane that we presented in [16]. The optimized code

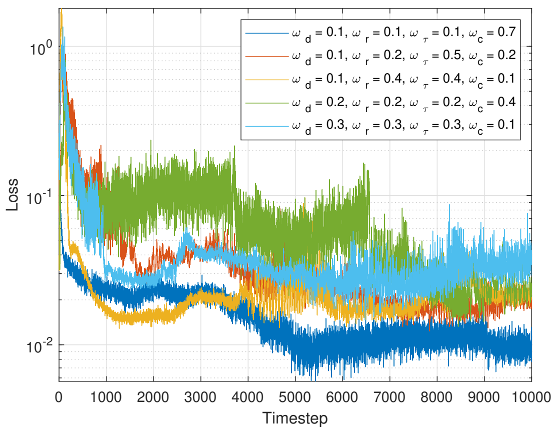

Fig. 4. Convergence of loss function for different weight settings of sub-reward function.

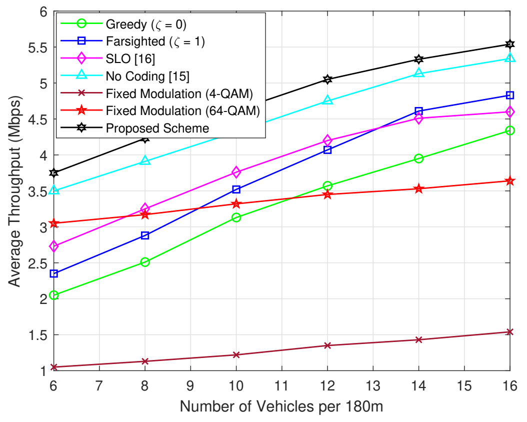

Fig. 5. Comparison of average throughput versus number of vehicles for different approaches under consideration.

rate and modulation order are then applied to all other links (vehicles behind). In this scheme, the states of other links are assumed to be unknown to the agent since only the observed single link (vehicle in the same lane) is tracked. As a result, we named it the single link optimization (SLO) scheme.

*4) **Greedy**:* This method is a variant of our scheme, where we set the discount factor to $\zeta = 0$ in (15), while keeping all other parameters of the systems as reported in Table IV. In this scenario, the agent selects an action that maximizes only the immediate reward.

*5) **Farsighted**:* This method is another variant of our scheme in which we assume the discount factor to be $\zeta = 1$ in (15). In this scheme, we maintain all other parameters of the systems to be the same as reported in Table IV. This scheme emphasizes future rewards while disregarding immediate rewards.

*6) **Fixed Modulation (4-QAM)**:* This is a variant of our proposed scheme. In this scheme, instead of an adaptive modulation scheme, we consider a fixed modulation, namely 4-QAM. We keep other optimization parameters the same with the proposed scheme. Here, we optimize the code rates and vehicle's speed while keeping the modulation scheme fixed.

*7) **Fixed Modulation (64-QAM)**:* This is another variant of the fixed modulation scheme in which we use 64-QAM as modulation. The only difference with 4-QAM is the modulation order we use.

C. Performance Evaluation

We first perform an ablation study of different weight values of reward function (10) to select the setting that leads to faster convergence of the loss function. In particular, we present five different settings weight values of distance, ω_d; BER, ω_r; latency, ω_τ and rate, ω_c; for visualization simplicity in Fig. 4 though more settings could be illustrated. From the figure, we see that the setting $\omega_d = 0.1$, $\omega_r = 0.1$, $\omega_\tau = 0.1$, $\omega_c = 0.7$ provides faster convergence and leads to lower loss values. Therefore, we employ this setting for the rest of our evaluation.

In the following analysis, we investigate how the proposed algorithm impacts various performance parameters, such as throughput, latency, and reliability for the schemes under comparison. To begin with, we evaluate the effect of the density of vehicles on the average throughput and average latency for the schemes under comparison by varying the number of vehicles between 6 to 16 per 180 meters (m). Fig. 5 shows the average throughput results for the different schemes, and it is evident that the proposed scheme outperforms the other schemes across all ranges of vehicles, showcasing significantly higher average throughput. From the figure, it is evident that No Coding scheme offers the second-best performance, which shows the advantage of employing the 5G-NR codes and optimizing their code rate. Whereas the performance gap between the proposed scheme and the SLO scheme is larger than that with the No Coding scheme. This is because, in the SLO scheme, the state of one link is observed while optimizing the parameters of other links based on the policy of the observed link, which can lead to constraints not being satisfied, resulting in a larger performance gap from the proposed scheme. For farsighted and greedy schemes, the gap grows further because of considering only future rewards and immediate rewards, respectively. Additionally, it can be seen from the figure that fixed modulation schemes achieve a relatively fixed throughput, and are not able to take advantage of increased vehicle density because there is no change in the modulation scheme. Among these, 4-QAM has the lowest average throughput. These results demonstrate the advantages of using adaptive modulation.

We present the average latency for the different schemes compared at varying numbers of vehicles in Fig. 6. From the figure, we observe that the proposed scheme outperforms all other schemes, but the 64-QAM with which has comparable performance. Similar to the average throughput in Fig. 5, the 4-QAM and 64-QAM offer almost fixed average latency at all the range of vehicles. It is also apparent from the figure that the 64-QAM has lower average latency than the proposed scheme until the number of vehicles is 12. This is because, at fixed modulation, we can achieve lower latency at the cost of violating other constraints, which we will discuss in the next paragraph. Moreover, the No Coding scheme shows the lowest latency gap and hence, we can see the effect of code rate optimization in our proposed scheme. Similarly, the SLO scheme has a larger average latency gap with the proposed scheme than the No Coding scheme, and it even has higher

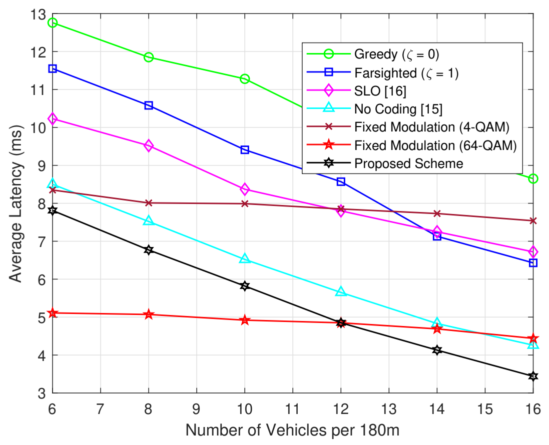

Fig. 6. Comparison of average latency versus number of vehicles with different considered schemes.

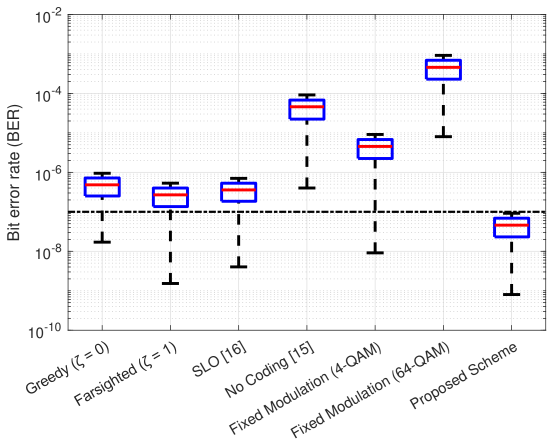

Fig. 7. Box plot showing the maximum and minimum BER offered for all the schemes under comparison.

latency when the number of vehicles exceeds 14 per 180m. Therefore, we can conclude that we can achieve a higher rate and lower latency by using code rate optimization and multi-link policy maximization like in the proposed scheme.

We, then, visualize how the proposed scheme satisfies the uRLLC requirements while maximizing the throughput by analyzing the BER and latency performance. In this paper, we set the requirements for ultra-reliability to meet the BER of 10^{-7} and low latency to satisfy 10 ms latency. We begin by showing the boxplot for the BER performance in Fig. 7 to evaluate whether the reliability requirement is met by all the schemes. From the evaluation, we can observe that most BER values are clustered near the maximum and are considerably distant from the minimum for all schemes. This is because our proposed scheme consistently maintains safe distances between vehicles to avoid collisions and prevent them from coming too close. Moreover, in this analysis, we only examine the maximum observed BER of all available links at each time slot for each scheme. We have also plotted a reference line for the BER constraint of 10^{-7} (dashed black line). From the figure, it can be observed that the proposed algorithm always satisfies the reliability requirement, while the other schemes fail to meet the constraint most of the time. In particular, the No Coding scheme cannot satisfy the BER requirement. Therefore, it is evident that channel coding is necessary to achieve the reliability requirements. The SLO scheme can respect the BER for more than half of the time because we optimize the performance of multiple links based on the observation of single link parameters. In this scenario, there is the possibility of bad policies for other vehicles, which was considered good for the observed link. For fixed modulations, 4-QAM can meet the BER constraint less than half of the time, whereas 64-QAM can never satisfy them. In particular, 64-QAM has the lowest BER performance of any of the comparison schemes. This occurs because we need to transmit data at a higher rate all the time, which makes it difficult to guarantee reliability over a longer distance. Conversely, 4-QAM can transmit and receive data reliably most of the time.

Finally, Fig. 8 illustrates the boxplots of the observed latency to examine whether the low latency requirement (10 ms) is met by the different comparison schemes. We train the models for 10000 timesteps. Similarly to Fig. 7 where we explored BER performance, we have also computed the maximum observed latency of all available links at the current time slot. We have also drawn a reference line for the latency constraint (dashed black line) in Fig. 8. From the figure, it can be seen that the proposed and fixed modulation schemes consistently meet the low latency requirements of 10 ms, while the other schemes fail to meet this constraint most of the time. Specifically, for the greedy, farsighted, SLO, and No Coding schemes, the maximum observed latency is 15.5 ms, 13.8 ms, 11.5 ms, and 11 ms, respectively. The 4-QAM and 64-QAM achieve lower latency since we use fixed modulation while sacrificing the throughput and BER.

The above performance comparisons demonstrate that our proposed vehicular OCC system can maximize throughput while ensuring uRLLC, while the other schemes, we compare with, cannot meet both the reliability and delay requirements or none. Although fixed modulation schemes, 4-QAM and 64-QAM can satisfy the latency requirements at the expense of lower average throughput, they violate reliability requirements. It is evident from the preceding that we achieve better performance in the multi-agent vehicular OCC system by performing code rate optimization and utilizing adaptive modulation schemes.

VI. Conclusion

In this paper, we investigate a multi-link DRL-based throughput maximization scheme to ensure uRLLC in vehicular OCC system. To accomplish this, we choose the optimal code rate, modulation scheme and speed of vehicles for multiple vehicular links. To meet ultra-reliability requirements, 5G NR LDPC codes and an adaptive modulation scheme are used. We then solve the continuous optimization problem using an actor-critic algorithm through the Wolpertinger policy for multiple links. We compare the proposed scheme's performance to that of various variants of our scheme. According to the results of the evaluation, the proposed method achieves significantly higher average throughput and lower latency than all of the schemes under consideration. The results also show that our scheme always meets uRLLC requirements, whereas other schemes fail to meet the majority of the time. This happens

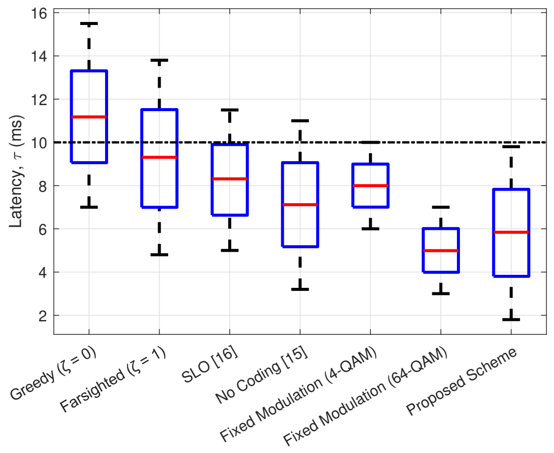

Fig. 8. Box plot showing the maximum and minimum latency offered for different schemes under comparison.

because we optimize the parameters (finding the best code rate, modulation, and vehicle distance) for multiple vehicular links at the same time. While schemes that do not use channel codes and use fixed modulation cannot guarantee reliability. The reliability and low latency requirements cannot be met for single link optimization because the agent optimizes policies for other links without considering their observed states, resulting in a sub-optimal solution most of the time.

References

[1] G. Araniti *et al.*, "LTE for vehicular networking: A survey," *IEEE Commun. Mag.*, vol. 51, no. 5, pp. 148–157, May 2013.

[2] S. Nayak and S. Roy, "Novel markov chain based URLLC link adaptation method for 5G vehicular networking," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 12 302–12 311, Oct. 2021.

[3] X. Ge, "Ultra-reliable low-latency communications in autonomous vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5005–5016, Mar. 2019.

[4] I. Takai *et al.*, "Optical vehicle-to-vehicle communication system using LED transmitter and camera receiver," *IEEE Photon. J.*, vol. 6, no. 5, pp. 1–14, Oct. 2014.

[5] P. Luo *et al.*, "Experimental demonstration of RGB LED-based optical camera communications," *IEEE Photon. J.*, vol. 7, no. 5, pp. 1–12, Oct. 2015.

[6] H. Gamage, N. Rajatheva, and M. Latva-Aho, "Channel coding for enhanced mobile broadband communication in 5G systems," in *Proc. 2017 EuCNC*, Oulu, Finland, Jun. 2017, pp. 1–6.

[7] M. Arabaci *et al.*, "High-rate nonbinary regular quasi-cyclic LDPC codes for optical communications," *J. Lightwave Technol.*, vol. 27, no. 23, pp. 5261–5267, Aug. 2009.

[8] T. Richardson and S. Kudekar, "Design of low-density parity check codes for 5G new radio," *IEEE Commun. Mag.*, vol. 56, no. 3, pp. 28–34, Mar. 2018.

[9] A. K. Sangaiah *et al.*, "LACCVoV: linear adaptive congestion control with optimization of data dissemination model in vehicle-to-vehicle communication," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 5319–5328, Dec. 2020.

[10] L. Liang, H. Ye, and G. Y. Li, "Spectrum sharing in vehicular networks based on multi-agent reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2282–2292, Aug. 2019.

[11] Z. Du, C. Wang, Y. Sun, and G. Wu, "Context-aware indoor VLC/RF heterogeneous network selection: Reinforcement learning with knowledge transfer," *IEEE Access*, vol. 6, pp. 33 275–33 284, Jun. 2018.

[12] H. Yang *et al.*, "Learning-based energy-efficient resource management by heterogeneous RF/VLC for ultra-reliable low-latency industrial IoT networks," *IEEE Trans. Industr. Inform.*, vol. 16, no. 8, pp. 5565–5576, Aug. 2020.

[13] G. Dulac-Arnold *et al.*, "Deep reinforcement learning in large discrete action spaces," *arXiv preprint arXiv:1512.07679*, 2015.

[14] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971v6*, Jul. 2019.

[15] A. Islam, N. Thomos, and L. Musavian, "Multi-agent deep reinforcement learning for spectral efficiency optimization in vehicular optical camera communications," *IEEE Trans. Mob. Comput.*, vol. 23, no. 5, pp. 3666–3679, May 2024.

[16] A. Islam, N. Thomos, and L. Musavian, "Achieving uRLLC with machine learning based vehicular OCC," in *Proc. IEEE GLOBECOM*, Rio de Janeiro, Brazil, Dec. 2022, pp. 1–6.

[17] G. Vidyarthi, A. Ngom, and I. Stojmenovic, "A hybrid channel assignment approach using an efficient evolutionary strategy in wireless mobile networks," *IEEE Trans. Veh. Technol.*, vol. 54, no. 5, pp. 1887–1895, Nov. 2005.

[18] A. Koller and S. D. Noble, "Domination analysis of greedy heuristics for the frequency assignment problem," *Discrete Math.*, vol. 275, no. 1-3, pp. 331–338, Jan. 2004.

[19] I. Takai, *et al.*, "LED and CMOS image sensor based optical wireless communication system for automotive applications," *IEEE Photon. J.*, vol. 5, no. 5, pp. 6 801 418–6 801 418, Oct. 2013.

[20] Y. Goto *et al.*, "A new automotive VLC system using optical communication image sensor," *IEEE Photon. J.*, vol. 8, no. 3, pp. 1–17, Jun. 2016.

[21] H. Ye, G. Y. Li, and B. H. F. Juang, "Deep reinforcement learning based resource allocation for V2V communications," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3163–3173, Apr. 2019.

[22] I. Lee and D. K. Kim, "Decentralized multi-agent DQN-based resource allocation for heterogeneous traffic in V2X communications," *IEEE Access*, vol. 12, pp. 3070–3084, Jan. 2024.

[23] J. Xue *et al.*, "Cooperative deep reinforcement learning enabled power allocation for packet duplication URLLC in multi-connectivity vehicular networks," *IEEE Trans. Mob. Comput.*, vol. 23, no. 8, pp. 8143–8157, Aug. 2024.

[24] A. Islam, M. T. Hossan, and Y. M. Jang, "Convolutional neural network scheme–based optical camera communication system for intelligent internet of vehicles," *Int. J. Distrib. Sens. Netw.*, vol. 14, no. 4, pp. 1–15, Apr. 2018.

[25] S. A. I. Alfarozi *et al.*, "Square wave quadrature amplitude modulation for visible light communication using image sensor," *IEEE Access*, vol. 7, pp. 94 806–94 821, Jul. 2019.

[26] M. Bennis, M. Debbah, and H. V. Poor, "Ultrareliable and low-latency wireless communication: Tail, risk, and scale," *Proc. IEEE*, vol. 106, no. 10, pp. 1834–1853, Sep. 2018.

[27] 3GPP, "Study on scenarios and requirements for next generation access technologies," *Technical Specification Group Radio Access Network, Technical Report 38.913*, 2016.

[28] D. A. Plaisted, "Some polynomial and integer divisibility problems are NP-HARD," in *Proc. 17th SFCS*, Houston, TX, USA, Oct. 1976, pp. 264–267.

[29] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: Algorithms and complexity*. Courier Corporation, 1998.

[30] R. S. Sutton and A. G. Barto, *Introduction to reinforcement learning*. MA, USA: MIT press Cambridge, 1998, vol. 135.

[31] T. Zinchenko, "Reliability assessment of vehicle-to-vehicle communication," doctoralthesis, Technische Hochschule Wildau, 2014.

[32] D. Silver *et al.*, "Deterministic policy gradient algorithms," in *Proc. 2014 ICML*, Beijing, China, Jun. 2014, pp. 387–395.

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

[34] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the Brownian motion," *Phys. Rev.*, vol. 36, no. 5, pp. 823–841, Sep. 1930.

[35] C. Zhong, M. C. Gursoy, and S. Velipasalar, "Deep reinforcement learning-based edge caching in wireless networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 1, pp. 48–61, Jan. 2020.

[36] N. Papernot and P. McDaniel, "Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning," *arXiv preprint arXiv:1803.04765*, 2018.

[37] H. Maei *et al.*, "Convergent temporal-difference learning with arbitrary smooth function approximation," in *Proc. 22nd NIPS'09*, vol. 22, NY, United States, Dec. 2009, pp. 1204 – 1212.

[38] D. A. Guastella and G. Bontempi, "Traffic modeling with SUMO: A tutorial," *arXiv preprint arXiv:2304.05982*, 2023.

[39] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[40] J. H. Bae *et al.*, "An overview of channel coding for 5G NR cellular communications," *APSIPA Trans. Signal Inf. Process.*, vol. 8, Jun. 2019.

[41] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

Amirul Islam received his PhD degree in Computing and Electronic Systems from the University of Essex, UK, in 2022. He completed his MSc from Kookmin University, South Korea. He currently serves as an Assistant Professor in the Department of Electrical and Electronic Engineering at the American International University-Bangladesh, Bangladesh. Prior to this, he held the position of a Post-Doctoral Researcher at the Visual Artificial Intelligence Laboratory, Oxford Brookes University, UK. His research interests include machine learning for communication, optical camera communication, deep reinforcement learning, automotive vehicular communications, and optimization strategies.

Nikolaos Thomos is currently a Professor at the University of Essex, U.K. Before that, he was a Senior Researcher at the Ecole Polytechnique Federale de Lausanne (EPFL) and the University of Bern, Switzerland. He received the Diploma (MSc equivalent) and Ph.D. degrees from the Aristotle University of Thessaloniki, Greece. He has been PI and co-I for a number of ESPRC, SNSF, EC, InnovateUK, and Hassler foundation-funded projects. His research interests include machine learning for communications, multimedia communications, semantic communications, design of new waveforms, cross-layer optimization, network coding, information-centric networking, source and channel coding, and signal processing. He is an elected member of the IEEE MMSP Technical Committee (MMSP-TC) for the period 2019-2024. He received the highly esteemed Ambizione Career Award from the Swiss National Science Foundation (SNSF).

Leila Musavian is a Professor at University of Essex. Previously, she was Deputy Pro-Vice-Chancellor for Research at University of Essex and Reader in Telecommunications at the School of Computer Science and Electronic Engineering. Prior to that, she was Lecturer/Senior Lecturer at InfoLab21, Lancaster University, UK, Research Associate at McGill University, Canada, Research Associate at Loughborough University, UK and Post-Doctoral Fellow at INRS-EMT, University of Quebec, Canada.

Her research interests lie in Radio Resource Management for 6G/B5G communications, low latency communications, ML for Communications, NFC, NTN and Energy Harvesting Communications. She is currently editor of the IEEE Communications Surveys and Tutorials (COMST) and has previously been Editor of IEEE TRANSACTIONS OF WIRELESS COMMUNICATIONS. She was the Conference Workshop Co-Chair of VTC-Spring-2020, the Wireless Communications Symposium Leading Co-Chair for IEEE ICC 2021, Montreal, Canada, conference TPC Co-Chair of IEEE CAMAD 2021, Portugal and Track 2 Lead chair of IEEE WCNC 2023. She is the founding chair and co-chair of the IEEE UK & Ireland Future Networks local group.