Lyapunov-Guided Offloading Optimization Based on Soft Actor-Critic for ISAC-Aided Internet of Vehicles

Yonghui Liang, Huijun Tang, *Member, IEEE*, Huaming Wu, *Senior Member, IEEE*, Yixiao Wang, and Pengfei Jiao, *Member, IEEE*

***Abstract*—Due to numerous computation-intensive and delay-sensitive tasks in the Internet of Vehicles (IoV), Vehicular Edge Computing (VEC) is increasingly playing a crucial role as a key solution in the IoV. However, how to concurrently enhance communication quality and reduce the cost of latency and energy has emerged as a critical challenge in VEC. To tackle the above problem, we propose a Lyapunov-guided offloading based on the Soft Actor-Critic (SAC) algorithm, named LySAC, to minimize the average cost of the Integrated Sensing and Communications (ISAC) technology-aided IoV, where ISAC technology can effectively improve the communication quality by harnessing high-frequency waveforms to seamlessly integrate communication and sensing functionalities. First, we model the offloading process of ISAC-Aided IoV as an optimization problem of the joint cost of delay and energy with long-term energy consumption and queue stability. Then we formulate the optimization problem as a Lyapunov optimization and utilize the SAC method to find the optimal offloading decisions. Finally, we conduct extensive experiments and the results demonstrate the effectiveness and superiority of the proposed LySAC in minimizing total cost while maintaining queue stability and meeting long-term energy requirements compared with other several baseline schemes.**

***Index Terms*—Computation offloading, integrated sensing and communications, Lyapunov optimization, vehicular edge computing.**

I. Introduction

INTERNET of Vehicles, a subset of the broader domain of the Internet of Things (IoT), revolve around the connectivity and communication among vehicles, enabling seamless data transmitted between vehicles, infrastructure, and other devices to improve road safety, traffic efficiency, and overall transportation effectiveness [1]. However, the continuous influx of data, much of which is time-sensitive, manifests as a flood of computational tasks that overwhelm the limited processing capabilities of vehicles, thereby increasing the risks of on-road incidents. This means that the numerous time-sensitive tasks constitute a contradiction with the limitation of computational resources of the Internet of Vehicles (IoV). A promising solution to the contradiction is Vehicular Edge Computing (VEC) [2], [3], which offloads tasks from vehicles to Road-Side Units (RSUs) positioned along the roads, where the RSUs have advanced computational capabilities surpassing those of the vehicles [4].

As an extension and application of edge computing, VEC offers a viable solution to the computational challenges in IoV. This not only alleviates the computational burden on vehicles but also significantly reduces the latency in data processing, which is crucial for time-sensitive applications such as autonomous driving and real-time traffic management [5]. The integration of VEC into IoV also enables advanced applications like dynamic route planning, real-time video analytics, and environmental sensing [6], all of which require substantial computational resources and quick response times that are not feasible through conventional cloud computing. Additionally, VEC supports the aggregation and preprocessing of data at the edge, which leads to reduced network congestion and improved data privacy [7]. Given these advantages, VEC stands as a cornerstone in the advancement of intelligent transportation systems, shaping a more efficient, safe, and smart vehicular environment [8].

Meanwhile, Integrated Sensing and Communication (ISAC) is an emerging paradigm that integrates wireless communication and radar sensing functionalities into a unified framework by the seamless amalgamation of sensor technologies with communication systems to enhance data acquisition and transmission capabilities [9]. This approach is particularly pivotal for the development of the sixth generation (6G) networks [10], which aim to revolutionize the telecommunication landscape by offering ultra-high-speed, low-latency, and massive connectivity, which supports a wide range of applications, from autonomous vehicles to smart cities [11]. By employing techniques such as using a shared waveform or implementing frequency division multiplexing and time division multiplexing, devices can leverage the capabilities of ISAC [12]. The incorporation of IoV and ISAC, which is called ISAC-aided IoV, significantly boosts the efficiency of network communication computation, which makes vehicular network able to meet requirements for minimal latency, robust system stability, and heightened security [13].

However, in the context of ISAC-aided MEC networks, where tasks and communication modes have different performance requirements compared to ordinary scenarios, the applicability of the methods of previous studies may be limited. Especially in

Manuscript received 28 February 2024; revised 28 June 2024; accepted 14 August 2024. Date of publication 19 August 2024; date of current version 5 November 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62071327 and in part by Tianjin Science and Technology Planning Project under Grant 22ZYYYJC00020. Recommended for acceptance by X. Peng. *(Corresponding author: Huaming Wu.)*

Yonghui Liang, Huaming Wu, and Yixiao Wang are with the Center for Applied Mathematics, Tianjin University, Tianjin 301500, China (e-mail: liangyonghui@tju.edu.cn; whming@tju.edu.cn; wang_yixiao@tju.edu.cn).

Huijun Tang and Pengfei Jiao are with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China (e-mail: tanghuijune@hdu.edu.cn; pjiao@hdu.edu.cn).

Digital Object Identifier 10.1109/TMC.2024.3445350

vehicular networks, where demands for minimal latency, robust system stability, and heightened security prevail, the need for tailored solutions that can adapt to rapidly changing network topologies and handle tasks efficiently becomes critical.

To cope with the aforementioned challenges, we explore a sophisticated computation offloading and resource allocation strategy within ISAC-enhanced VEC networks, grounded in Lyapunov optimization and Soft Actor-Critic (SAC) methodologies. Lyapunov optimization ensures system stability and manages long-term constraints, such as energy consumption and queue stability. It excels in handling optimization problems over time, maintaining performance under fluctuating conditions. Conversely, the SAC method uses deep reinforcement learning to make near-optimal decisions in dynamic and stochastic VEC environments, balancing exploration and exploitation for real-time adaptation. Our algorithm integrates Lyapunov optimization's stability and constraint management with SAC's adaptive decision-making. This combination ensures long-term system stability while dynamically adjusting to environmental changes, significantly improving over algorithms that address these aspects separately.

Our primary objective is to intricately minimize the weighted sum of the system's latency and energy expenditure. The problem is initially reformulated by the Lyapunov optimization method to transfer the energy and queue stability constraints. Subsequently, we employ a SAC-based algorithm to determine the optimal decisions for offloading and resource allocation. The main contributions of this paper are outlined as follows.

- In the context of ISAC technology, this study introduces a novel ISAC-aided VEC computing framework and presents a visualization of the task processing flow in the central scheduling module of the edge server. Its primary function is to make strategic offloading decisions and allocate resources in a manner that optimally enhances the system's overall efficiency.
- This study introduces a minimization problem aiming at jointly optimizing the computation offloading and resource allocation decisions. It focuses on reducing the combined weight of system delay and energy consumption, a critical balance for efficient operation. To achieve this, the Lyapunov optimization method is employed to further transform the long-term queue stability and energy constraint, ensuring the queue stability and reducing overall computational complexity.
- This paper proposes a novel Lyapunov-guided offloading strategy based on the SAC algorithm called LySAC, which can devise efficient task scheduling and resource allocation strategies dynamically without relying on prior system state information. The strategy leverages both the Lyapunov transformation and the SAC-based algorithm to achieve its objectives.
- Extensive experimental results demonstrate that compared to several baseline schemes, the proposed LySAC algorithm can dynamically adjust decision variables to minimize the total system cost while maintaining queue stability and meeting long-term energy requirements.

The remainder of the paper is structured as follows: The related work is comprehensively investigated in Section II. We elaborate on the system model in Section III. The problem formulation is introduced in Section IV. Section V delves into the details of the Lyapunov optimization and the proposed SAC-based offloading and resource allocation approach. Simulation results are presented and discussed in Section VI. Finally, our conclusions are drawn in Section VII.

II. Related Work

A. ISAC-Aided V2X Network

In the context of ISAC-aided Vehicle-to-Everything (V2X) networks, which facilitate communication between a vehicle and any entity that may influence or be influenced by the vehicle, some scholars have already conducted research on system communication management and offloading decisions. For instance, Mu et al. [14] introduced a DNN approach to predict beamforming with an ISAC-Enabled network. Liu et al. [15] proposed a joint computation offloading and resource allocation strategy to build greener V2X networks with MEC and ISAC technologies. Li et al. [16] developed a Vehicle-to-Infrastructure (V2I) system that leverages ISAC signaling to enhance communication management between vehicles and road infrastructure. This system enables the tracking and prediction of vehicle motion. Bai et al. [17] suggested an approach to enhance the system's promptness and efficiency by maximizing the Age of Information (AoI) utility in ISAC-aided vehicular network, which has the potential to expedite the update process of sensing information by markedly decreasing AoI. Li et al. [18] presented an Ambient Backscatter Communication-aided ISAC system for V2X networks, addressing the challenge of coupling radar parameter estimation and signal demodulation through the innovative use of Backscatter Devices (BDs) and a 3D-Newtonized Orthogonal Matching Pursuit estimator. Huang et al. [19] proposed a MEC-assisted ISAC model featuring short-packet transmissions, where mutual information is utilized to evaluate radar sensing performance and assess both reliability and latency in processing radar data through edge computing. However, the aforementioned literature does not propose a VEC task computing framework with ISAC assistance and does not present a visualization of the task processing flow, nor does it adequately elaborate on this aspect.

B. DRL-Based Approaches

The dynamic nature of vehicular networks, characterized by their inherent high mobility, presents significant challenges, particularly in the realm of predicting network conditions and managing rapidly evolving computation and bandwidth demands. This complexity often leads to what is known as the curse of dimensionality. Traditional optimization techniques, mainly devised for static or slowly changing scenarios, fall short in addressing these complexities, primarily due to their inadequacy in adapting to the swiftly altering landscape of vehicular networks. In this context, the integration of Deep Reinforcement Learning

(DRL) with VEC emerges as a pivotal solution [28]. DRL is celebrated for its adaptive learning capabilities and proficiency in making near-optimal decisions based on environmental interactions, thereby excelling in scenarios where conventional algorithms falter [29].

The convergence of DRL and VEC opens new avenues for addressing the intricacies of resource allocation, latency mitigation, and seamless task offloading within the 6G vehicular landscape [30]. By utilizing DRL's capabilities in dynamic environments, this approach presents potential solutions to the challenges of high mobility and the evolving demands of vehicular networks. Tang et al. [20] utilized Double Deep Q-Network (DDQN) to design a Dynamic Framing Offloading algorithm, minimizing idle time by making offloading decisions as soon as a vehicle's subtasks are generated. Huang et al. [21] used the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm to design a task offloading strategy aware of shared task types and vehicle speed for resource allocation. The algorithm presented in [22] successfully minimizes the system's total latency, leveraging the capabilities of the SAC approach. Qu et al. [23] introduced a Deep Meta Reinforcement Learning-based Offloading algorithm that integrates multiple parallel DNNs with Q-learning for precise offloading decisions. This algorithm harnesses the comprehensive perception of deep learning, the strategic decision-making of reinforcement learning, and the accelerated environmental adaptation of meta-learning. However, the majority of current DRL-based strategies tend to focus on either compute-intensive tasks or those that are delay-sensitive, with only a handful of studies examining both dimensions concurrently.

C. Lyapunov Optimization-Based Approaches

The Lyapunov optimization technique is an illustrious mathematical construct in control theory [31] due to its ability to ensure system stability and foster long-term rewards. By leveraging the properties of the Lyapunov function, Lyapunov optimization is employed to encapsulate both system stability and performance metrics [32], [33]. Vehicular networks require algorithms that are computationally efficient and provide stable solutions due to their rapidly fast-changing characteristic. To meet these demands in rapidly changing vehicular networks, the application of the Lyapunov optimization algorithm has been explored [34]. Hung et al. [24] proposed a Mobile Edge Computing (MEC)-assisted task offloading method for vehicle platoons using Lyapunov optimization. They also developed a vehicle-centric framework employing these techniques to enhance low latency in Vehicle-to-Vehicle (V2V) networks, which facilitate direct communication between vehicles.

Except for combining with traditional optimization techniques such as the game theory [25], the Lyapunov optimization technique is often integrated with deep learning tools to facilitate offloading decisions and resource allocation strategies in recent works. For instance, Bi et al. [26] combines the advantages of Lyapunov optimization and DRL technique. The authors applied Lyapunov optimization to decouple the multi-stage stochastic mixed integer non-linear programming into deterministic subproblems. And then they utilized model-free DRL to solve the subproblems with very low computational complexity. What's more, Kumar et al. [22] proposed a Lyapunov-based multi-agent DRL method that jointly optimizes computing task distribution and radio resource allocation to minimize energy consumption and delay requirements. Li et al. [27] introduced a joint radio and resource allocation algorithm that optimizes the average transmission power minimization problem in VEC systems, taking into account the Quality of Service (QoS) requirements of tasks and the impact of time-varying channels. However, the efficacy of aforementioned strategies using Lyapunov optimization is compromised due to their difficulty in accurately assessing queue backlogs and real-time network conditions, a situation exacerbated by not incorporating the latest deep learning algorithms, thereby limiting their adaptability and optimization in dynamic network environments.

D. Comparison of Selected Related Studies

Table I provides a detailed comparison of our study with existing research in the domain of MEC and vehicular networks. This comparison covers several critical dimensions, including the scenarios considered (e.g., VEC and V2X), the incorporation of ISAC-aided techniques, the types of DRL algorithms employed, the primary objectives of each study, and the proposed decision-making strategies.

As shown in Table I, we are the first to combine the Lyapunov optimization theory and SAC algorithm in an ISAC-assisted vehicular network scenario for resource allocation and offloading decisions. This approach effectively reduces the total system cost while ensuring queue stability. In contrast, other related works either do not involve ISAC-assisted vehicular networks or do not employ Lyapunov optimization methods or reinforcement learning techniques. Additionally, our optimization objective simultaneously considers both delay and energy consumption, a dual objective that is not commonly addressed in the literature despite being critical for vehicular networks. Previous studies often focus on only one aspect or optimize for relatively secondary metrics. Notably, our study introduces a novel ISAC-aided VEC computing framework and provides a visualization of the task processing flow within the central scheduling module of the edge server, a feature that has not been addressed in other works.

III. System Model and Problem Formulation

A. ISAC-Assisted Vehicular Network Scenarios

In this context, we introduce a vehicular network scenario that integrates communication, sensing, and computation, and propose a VEC-assisted autonomous driving framework. As depicted in Fig. 1, we consider a two-way straight city road segment, where at regular intervals (dependent on the RSU communication range), there exists an ISAC-aided RSU. These RSUs, equipped with ISAC capabilities, can emit beams with both communication and sensing functionalities, and meanwhile are integrated with the MEC server, offering computational capacities surpassing those of mobile vehicles. Adjacent RSUs,

TABLE I
COMPARISON OF SELECTED RELATED STUDIES

Reference	Scenario	ISAC-Aided	Lyapunov-Guided	DRL Type	Objective	Decisions
[14]	VEC	✓	✗	DNN	Estimation performance	Angle estimates
[15]	VEC	✓	✓	DDQN	Queuing latency	Offloading decision+Resource allocation
[16]	V2X	✓	✗	-	Network overheads	Communication beam management
[17]	V2X	✓	✗	-	Age of Information	Sensing+Transmission duration
[18]	V2X	✓	✗	-	Coupling challenge	Radar parameter estimate
[19]	VEC	✓	✗	-	Energy	Packet parameters+Resource allocation
[20]	VEC	✗	✗	DDQN	Delay	Offloading decision
[21]	VEC	✗	✗	MADDPG	Energy+Revenue of tasks	Offloading decision+Resource allocation
[22]	VEC	✗	✓	MADDPG	Energy	Offloading decision+Ratio resource allocation
[23]	VEC	✗	✗	Meta RL	Energy+Delay	Offloading decision
[24]	V2V	✗	✓	-	Delay	Offloading decision+Resource allocation
[25]	MEC	✗	✓	-	Transfer cost+Delay	UAV's activity+Channel allocation
[26]	MEC	✗	✓	DNN	Data processing capability	Offloading decision+Resource allocation
[27]	MEC	✗	✓	-	Transmission energy	Resource allocation
Ours	**VEC**	✓	✓	**SAC**	**Energy+Delay**	**Offloading decision+Resource allocation**

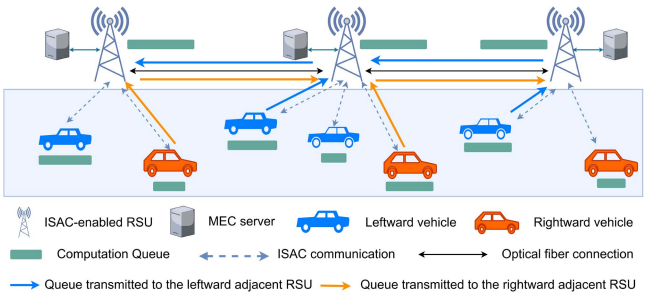

Fig. 1. ISAC-assisted vehicular network model.

interconnected through optical fibers, have the capability to transfer tasks–anticipated to be unfinished by vehicles exiting their range–to the neighboring RSU that the vehicles are approaching. The set of vehicles within an RSU's range is denoted as $\mathcal{V} = \{1, 2, \ldots, N\}$.

RSUs can sense their surroundings in real time. The sensed data undergoes preprocessing at the RSU side and is transmitted to neighboring RSUs via optical fibers, while also receiving real-time sensing information from these adjacent RSUs. This data at the RSU undergoes sensor information fusion, allowing a single RSU to acquire real-time conditions of multiple RSU ranges. After completing the sensing of the environment, the RSU manages platoon driving for all vehicles within its range through its communication functionality. One can envision that vehicles in platoon mode maintain consistent speeds over specific time intervals, which contributes to system stability, thereby enhancing the safety of the scenario.

Table II summarizes the key variables used in the system model.

B. ISAC-Aided VEC Computing Framework

Compared to the traditional edge computing framework, which simply integrates sensors, communicators, and computational layers, a new type of VEC computing framework integrating communication and sensing technologies (as shown in Fig. 2) is constructed. This framework enables multiple interactions with data at different levels, thereby enhancing data utilization efficiency and improving decision-making quality.

TABLE II
NOTATIONS AND THEIR DESCRIPTION

Symbol	Description
ISAC	Integrated Sensing and Communication
$\mathcal{V}$	The vehicle set
σ	Duration of the time slot
$\phi_i(t)$	Task of vehicle i at time slot t
$\phi^e(t)$	Task of edge server at time slot t
$d_i(t)$	Data size of the task
c_i	Number of CPU cycles required for processing one bit
$T_i^{max}(t)$	The maximum tolerated delay of the task
$x_i(t)$	Offloading decision
$T_i^l(t)$	The local processing delay in vehicle i
$T_i^O(t)$	The total delay of offloading
$T_i(t)$	The total delay of task
$E_i^l(t)$	The local processing energy consumption in vehicle i
$E_i^{trans}(t)$	The transmission energy consumption
$E_i(t)$	The total energy consumption of task
$f_i^l(t)$	Local computing capacity of vehicle$i$
$f_i(t)$	Computational resource allocated to vehicle$i$ by RSU
$f^e(t)$	Computational resource for processing its own tasks
f^{max}	The maximum computational resource of RSU
$\pi_i^{max}(t)$	The maximum permissible latency for the vehicle's task
$\pi^e(t)$	The maximum permissible latency for the RSU's task
l	Path loss exponent
CG	Communication gains
V	Weight balancing the Lyapunov drift and the cost function
α	Weight balancing delay and energy
β	Normalization coefficient
γ	Discount factor in reinforcement learning
κ	Computation energy efficiency coefficient of vehicle k
$r_i(t)$	Transmission rate
$P_i(t)$	Transmission power
P_i^{max}	The maximum transmission power
$g_i(t)$	Channel power gain
N_0	Gaussian white noise
W	Channel bandwidth between vehicle and RSU

Task data from the application layer can be transmitted to the central dispatch module of the computation layer in the edge server via ISAC signals. This central scheduling module uses different algorithms to process information thoroughly and at multiple levels, making the best decisions for offloading and resource allocation to improve the system's efficiency.

The central scheduling module, as shown in Fig. 2, receives two types of information:

- The ISAC information resides within its own range, including traditional radar echoes from the sensing layer,

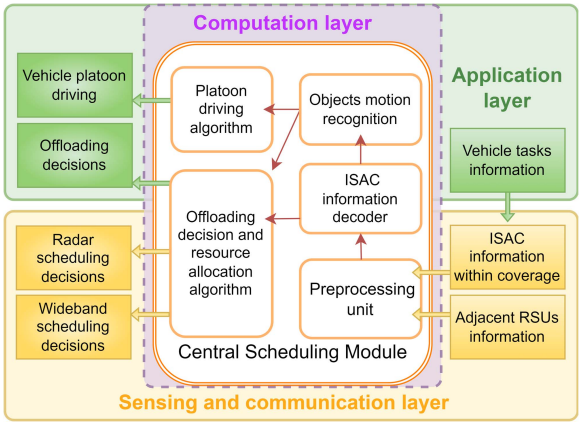

Fig. 2. ISAC-aided VEC computing framework.

communication messages, terminal information, other collaborative endpoint information, and task details sent by mobile applications.

- The information from adjacent RSUs, including traffic conditions, vehicle movements, and computational tasks or results submitted by vehicles about to enter this RSU from an adjacent one.

Initially, information from various sources is transmitted into the processing unit to undergo necessary preprocessing. Moreover, due to vehicles moving at a constant speed, computational tasks submitted by vehicles within range are assessed through simple distance calculations to determine if they can be completed before the vehicle exits the range. If not, these tasks will be transferred to an adjacent RSU that the vehicle is approaching. For ISAC data, after preprocessing, decoders are used to separate communication signals from radar sensing signals. Essential sensing and communication information are then fed into the object motion recognition module and the joint offloading decision and resource allocation module, respectively. The object motion recognition module identifies specific objects and their movement information from radar echoes, then conveys necessary details to the platoon driving module for vehicular formation driving control and management within the RSU range.

In offloading decision-making and resource allocation, a joint scheduling algorithm based on SAC considers the current communication status, vehicular terminal conditions, computational task information, and real-time computational resources. This algorithm determines the final offloading decisions and allocates bandwidth and radar communication power accordingly.

C. Communication and Computing Model

We adopt a time-slot computational model where the total duration is divided into T equal time slots, and the duration of each time slot is denoted as σ. This leads to the formulation of a discrete time-slot model, $\mathcal{T} = \{1, 2, \ldots, M\}$. At the commencement of each time slot, every vehicle generates a computational task, i.e.,

$$\phi_i^v(t) = \{d_i(t), c_i, \pi_i^{max}(t)\}, \tag{1}$$

where $d_i(t)$ indicates the data volume of the task for vehicle i in the tth time slot, and c_i denotes the number of CPU cycles necessary to process a single bit of data by vehicle i. This value varies between local and edge processing due to differences in device construction and performance. $\pi_i^{max}(t)$ represents the maximum permissible latency for the task. Concurrently, given that the edge RSU integrates communication-sensing-computation functionalities, it also generates respective computational tasks denoted as $\phi^e(t) = \{d^e(t), c^e, \pi^e(t)\}$.

In this paper, we employ a binary offloading strategy, with the offloading decision represented as $x_i(t) \in \{0, 1\}$. Specifically, $x_i(t) = 0$ indicates that the task is executed locally, while $x_i(t) = 1$ suggests that the task is offloaded for execution at the RSU.

1) Sensing-Enhanced Communication Model: In the context of Integrated Sensing and Communication (ISAC), the collaborative benefits between communication and sensing functions are evident. Scholars in the field have explored the synergistic gains between communication and sensing. Broadly, the synergy between communication and sensing can be categorized into two types: sensing-enhanced communication and communication-enhanced sensing. Existing research [35], [36] on the synergistic benefits of communication and sensing in ISAC often propose specific technical solutions on a case-by-case basis. However, there is a lack of comprehensive characterization methods for performance improvement resulting from the collaboration between communication and sensing. Therefore, performance enhancement in this area remains an open question. In [6], a framework for mmWave communication radar cooperative systems is presented. This vehicle network utilizes on-board mmWave radars and techniques like frequency division multiplexing to perform sensing and communication simultaneously. As the communication frequency band advances, the gains in both sensing and communication effectiveness will increase.

Let CG (communication gains) represent the ratio of the communication rate under sensing-assisted conditions to the communication rate under non-assisted conditions.

$$CG = \frac{R_{sen}}{R}, \tag{2}$$

where R_{sen} is the communication ratio enhanced by sensing, and R is the communication ratio without the enhancement of the sensing.

Let $r_i(t)$ be the data transmission rate between the vehicle and RSU:

$$r_i(t) = B \log_2 \left(1 + \frac{P_i(t) g_i(t)}{N_0}\right), \tag{3}$$

where B represents the communication bandwidth between the vehicle and RSU, $P_i(t)$ is the transmission power, N_0 is the Gaussian white noise power, and $g_i(t)$ is the channel gain. It's assumed that the channel gain remains constant within a time slot. The channel gain $g_i(t) = D_i^{-l}$ is defined, where $-l$ is the path loss exponent and D_i is the distance between vehicle i and the RSU.

Therefore, the ISAC-enhanced communication rate can be expressed as follows:

$$R_i(t) = CG * r_i(t). \tag{4}$$

2) Local Processing Model: In this case, let $T_i^l(t)$ be the local execution time, and $E_i^l(t)$ represents the local execution energy.

$$T_i^l(t) = \frac{d_i(t)c_i}{f_i^l(t)}, \tag{5}$$

$$E_i^l(t) = \kappa t_i^l(t) \left(f_i^l(t)\right)^3, \tag{6}$$

where κ is the energy consumption parameter for computing, and $f_i^l(t)$ represents the computing capability of vehicle i at the t^{th} time-slot.

3) Edge Processing Model: First, concerning the computational tasks generated at the edge, let $f^e(t)$ denote the computing resources allocated by the edge server for itself in time slot t. The required time to complete the task $\phi^e(t)$ is given by:

$$t^{RSU}(t) = \frac{d^e(t)c^e}{f^e(t)}. \tag{7}$$

Given that the data volume size of computation results is typically smaller than that of uploading the computing task, and the downlink data transmission rate is generally faster than the uplink data transmission rate, for simplification, the downlink transmission delay of returning the computation results can be neglected.

Let the task upload latency for offloading to RSU be $T_i^{up}(t)$, $T_i^e(t)$ be the expected edge computation latency, and $E_i^{up}(t)$ be the energy consumption during task upload. These are represented by the following:

$$T_i^{up}(t) = \frac{d_i(t)}{R_i(t)}, \tag{8}$$

$$T_i^e(t) = \frac{d_i(t)c_i}{f_i(t)}, \tag{9}$$

$$E_i^{up}(t) = \frac{d_i(t)}{R_i(t)} P_i(t), \tag{10}$$

where $f_i(t)$ is the computing resource allocated to vehicle i at the RSU's edge server.

D. Queuing Models

Each vehicle is equipped with two queues: one for tasks awaiting local computation and another for tasks pending offloading to the RSU. At the RSU, a designated queue is set up for each vehicle to hold tasks that are in line for processing.

- **Local Queue**

$$Q_i^L(t+1) = \max\left\{Q_i^L(t) - \frac{f_i^l(t)\sigma}{c_i} + (1 - x_i(t))\, d_i, 0\right\}, \tag{11}$$

where c_i is the CPU cycle number of processing a bit of data.

- **Local Waiting Queue**

$$Q_i^O(t+1) = Q_i^O(t) - B_i(t) + x_i(t)d_i(t), \tag{12}$$

where $B_i(t)$ is the data size offloaded from the vehicle i to RSU in time slot t. $B_i(t)$ can be denoted by:

$$B_i(t) = \min\left\{Q_i^O(t), r_i(t)\sigma\right\}. \tag{13}$$

- **Edge Queue**

$$Q_i^E(t+1) = \max\left\{Q_i^E(t) - \frac{f_i(t)\sigma}{c_i} + B_i(t), 0\right\}. \tag{14}$$

Drawing upon Little's Law, under steady-state conditions, the average queue length L in the system is equivalent to the average arrival rate λ multiplied by the average waiting time.

The queuing delay for tasks offloaded to the edge can be expressed as:

$$T_i^{eq}(t) = \frac{\widetilde{Q_i^E}(t)}{\widetilde{B_i}(t)}, \tag{15}$$

where $\widetilde{Q_i^E}(t)$ denotes the moving time-averaged task queue length,

$$\widetilde{Q_i^E}(t) = \frac{1}{m} \sum_{j=t-m+1}^{t} Q_i^E(j). \tag{16}$$

Furthermore, $\widetilde{B_i}(t)$ represents the moving time-averaged task arrival rate:

$$\widetilde{B_i}(t) = \frac{1}{m} \sum_{j=t-m+1}^{t} B_i(t). \tag{17}$$

The local queueing delay when tasks are chosen to be processed locally is as follows:

$$T_i^{lq}(t) = \frac{\widetilde{Q_i^L}(t)}{\widetilde{A_i}(t)}. \tag{18}$$

Consequently, the overall delay and total energy consumption for the vehicle i during time slot t can be encapsulated as follows:

$$T_i(t) = (1 - x_i(t))\left[T_i^l(t) + T_i^{lq}(t)\right] + x_i(t)[T_i^{up}(t) + T_i^e(t) + T_i^{eq}(t)], \tag{19}$$

$$E_i(t) = (1 - x_i(t))\, E_i^l(t) + x_i(t) E_i^{trans}(t). \tag{20}$$

E. Problem Formulation

The system's utility function during the time slot t can be defined as:

$$U_i(t) = \alpha\beta T_i(t) + (1 - \alpha)E_i(t), \quad \alpha \in [0, 1], \tag{21}$$

where α represents the weights assigned to delay and energy consumption, and β acts as a normalization coefficient to equate the magnitudes of delay and energy consumption. The values of α can be adjusted according to different types of tasks. For instance, if $\alpha = 1$, it signifies that the task is time-sensitive. Extending this, let $U(t) = \sum_{i=1}^{N} U_i(t)$.

We assume that x, f and P represent the offloading decision, computational resources, and communication power, respectively, as the three categories of decision variables. The

optimization problem can be formulated as follows:

$$\mathcal{P}_1: \min_{x,f,P} \lim_{M\to\infty} \frac{1}{M} \sum_{t=1}^{M-1} \mathbb{E}\left[U(t)\right] \tag{22}$$

$$s.t.\ C_1\text{: } x_i(t) \in \{0,1\}, \forall i \in \mathcal{V}, \forall t \in \mathcal{T} \tag{22a}$$

$$C_2\text{: } f^e(t) + \sum_{i=1}^{N} f_i(t) \le f^{max}, \forall t \in \mathcal{T} \tag{22b}$$

$$C_3\text{: } 0 \le f_i^l(t) \le f_i^{local\_max}(t), \forall i \in \mathcal{V}, \forall t \in \mathcal{T} \tag{22c}$$

$$C_4\text{: } 0 \le P_i(t) \le P_i^{max}, \forall i \in \mathcal{V}, \forall t \in \mathcal{T} \tag{22d}$$

$$C_5\text{: } 0 \le T_i^q(t) \le \mu, \forall i \in \mathcal{V}, \forall t \in \mathcal{T} \tag{22e}$$

$$C_6\text{: } 0 \le T^{RSU}(t) \le \sigma, \forall t \in \mathcal{T} \tag{22f}$$

$$C_7\text{: } \lim_{M\to\infty} \frac{1}{M} \sum_{t=1}^{M} \mathbb{E}[Q_i^L(t)] < \infty, \forall i \in \mathcal{V} \tag{22g}$$

$$C_8\text{: } \lim_{M\to\infty} \frac{1}{M} \sum_{t=1}^{M} \mathbb{E}[Q_i^E(t)] < \infty, \forall i \in \mathcal{V} \tag{22h}$$

$$C_9\text{: } \lim_{M\to\infty} \frac{1}{M} \sum_{t=1}^{M} \mathbb{E}[e_i(t)] \le e_t, \forall i \in \mathcal{V} \tag{22i}$$

where the constraint C_1 indicates that either a task is offloaded or locally executed, C_2 and C_3 define the maximum computational resource constraint respectively for the edge server and vehicle, C_4 specifies the maximum transmission power constraints for the vehicle, C_5 addresses the queuing latency limits for tasks, C_6 encapsulates the delay constraints for computation tasks on the edge side, C_7 and C_8 impose stability constraints on the queues at both the vehicle and edge server, ensuring that the queue lengths do not grow indefinitely over time, and C_9 pertains to the long-term average energy consumption limitations for the vehicle.

IV. Lyapunov-Guided Offloading Algorithm Based on the SAC

A. Problem Transformation With Lyapunov Optimization

We introduce the Lyapunov optimization method to transform the long-term delay constraint C_7, C_8 and energy consumption constraint C_9 to get the optimal computation offloading and resource allocation decision.

To address the average energy consumption constraint, a virtual queue representing energy consumption backlog in the vehicle at every instance is introduced, assumed to have an initial condition of $V_i(0) = 0$. The virtual queue's update process can be expressed as:

$$V_i(t+1) = \max\left\{V_i(t) - e(t) + E_i(t), 0\right\}, \forall i = 1, 2, \ldots, N \tag{23}$$

where $e(t)$ represents the data packet processed by the virtual queue in the time slot t and is i.i.d, while $e_i(t)$denotes the incoming data packet for the virtual queue. A high backlog value, $V_i(t)$, indicates that the energy consumed due to algorithmic decisions significantly exceeds the set long-term average.

Transforming the above equation:

$$V_i(t+1) - V_i(t) + e(t) \ge E_i(t). \tag{24}$$

Summing over t, we have:

$$\frac{V_i(M) - V_i(0)}{M} + e(t) \ge \frac{1}{M} \sum_{t=0}^{M-1} E_i(t). \tag{25}$$

Given that $V_i(0) = 0$, taking expectations on both sides:

$$\lim_{M\to\infty} \frac{\mathbb{E}[V_i(M)]}{M} + e(t) \ge \frac{1}{M} \sum_{t=0}^{M-1} \mathbb{E}[E_i(t)]. \tag{26}$$

Based on Lyapunov optimization theory, to ensure that the long-term average energy does not exceed its constraint, the following condition should hold:

$$\lim_{M\to\infty} \frac{\mathbb{E}[V_i(M)]}{M} = 0. \tag{27}$$

This implies that the constraint C_8 is inherently satisfied if the virtual queue $V_i(t)$ is mean-rate stable. The collection of virtual queues $V_i(t)$ is denoted as $V(t) \triangleq [V_1(t), V_2(t), \ldots, V_N(t)]$. Likewise, we have $Q^L(t) \triangleq [Q_1^L(t), Q_2^L(t), \ldots, Q_N^L(t)]$ and $Q^E(t) \triangleq [Q_1^E(t), Q_2^E(t), \ldots, Q_N^E(t)]$, with $\Theta(t) \triangleq \{V(t), Q^L(t), Q^E(t)\}$.

The Lyapunov function is defined as:

$$L\left(\Theta(t)\right) \triangleq \frac{1}{2}\left(\sum_{i=1}^{N} V_i(t)^2 + \sum_{i=1}^{N} Q_i^L(t)^2 + \sum_{i=1}^{N} Q_i^E(t)^2\right). \tag{28}$$

Introducing the Lyapunov drift function:

$$\Delta\left(\Theta(t)\right) \triangleq \mathbb{E}\left[L\left(\Theta(t+1)\right) - L\left(\Theta(t)\right) | \Theta(t)\right]. \tag{29}$$

Utilizing the Lyapunov-drift-penalty framework, we define the Lyapunov drift punishment function, and simultaneously solve for the minimum of the Lyapunov drift and the objective function:

$$\min \Delta_v(t) \triangleq \Delta\left(\Theta(t)\right) + V \cdot U(t), \tag{30}$$

where the weight V balances the relative significance of between $\Delta(\Theta(t))$ and $U(t)$. Consequently, we can dynamically construct an offloading strategy that balances the queue backlog, delay and energy consumption, adapting to real-time scenarios.

Next, in order to derive the upper bound for $\Delta_v(t)$, it is necessary to introduce Lemma 1.

Lemma 1: Let $Q(t+1)$, $Q(t)$, $a(t)$, and $b(t) \ge 0$, if $Q(t+1) = \max\{Q(t) - a(t) + b(t), 0\}$, then we have,

$$Q(t+1)^2 - Q(t)^2 \le 2Q(t)\left(b(t) - a(t)\right) + \left(b(t) - a(t)\right)^2.$$

The detailed proof is attached to Appendix A, available online. By applying Lemma 1 to the virtual energy queue mentioned in the paper, we have

$$V_i(t+1)^2 - V_i(t)^2 \le 2V_i(t)\left(E_i(t) - e(t)\right) + \left(E_i(t) - e(t)\right)^2. \tag{31}$$

Summing (31) over all vehicles, it holds

$$\frac{1}{2}\sum_{i=1}^{N}\left(V_i(t+1)^2 - V_i(t)^2\right) \le \frac{1}{2}\sum_{i=1}^{N}[2V_i(t)\left(E_i(t)-e(t)\right) + \left(E_i(t)-e(t)\right)^2]. \tag{32}$$

Then, for the sake of simplicity in subsequent expressions, we define

$$L\left(V(t)\right) \triangleq \frac{1}{2}\sum_{i=1}^{N} V_i(t)^2, \tag{33}$$

$$\Delta\left(V(t)\right) \triangleq \mathbb{E}\left[L\left(V(t+1)\right) - L\left(V(t)\right)|\Theta(t)\right]. \tag{34}$$

By applying the aforementioned definition to (33), it becomes evident that

$$L\left(V(t+1)\right) - L\left(V(t)\right) \le B_1(t) + \sum_{i=1}^{N} V_i(t)\left(E_i(t)-e(t)\right), \tag{35}$$

where $B_1(t) = \frac{1}{2}\sum_{i=1}^{N}(E_i(t)-e(t))^2$.

Lemma 2: Given that the vector $\Theta(t) \triangleq \{V(t), Q^L(t), Q^E(t)\}$ is the workload backlog status of queues, the Lyapunov drift function $\Delta(\Theta(t))$ can be controlled by an upper bound

$$\begin{aligned}\Delta\left(\Theta(t)\right) \le B &+ \sum_{i=1}^{N} V_i(t)\mathbb{E}\left[\left(E_i(t)-e(t)\right)|\Theta(t)\right] \\ &+ \sum_{i=1}^{N} Q_i^L(t)\mathbb{E}\left[\left(\left(1-x_i(t)\right)d_i(t) - \frac{f_i^l(t)\sigma}{c_i}\right)|\Theta(t)\right] \\ &+ \sum_{i=1}^{N} Q_i^E(t)\mathbb{E}\left[\left(B_i(t) - \frac{f_i(t)\sigma}{c_i}\right)|\Theta(t)\right],\end{aligned} \tag{36}$$

where $B = B_1 + B_2 + B_3$ and $B > 0$ is a finite constant.

The detailed proof is attached to Appendix B, available online. Then, according Lemma 2, we discern that:

$$\begin{aligned}\Delta_v(t) \le B &+ \sum_{i=1}^{N} V_i(t)\mathbb{E}\left[\left(E_i(t)-e(t)\right)|\Theta(t)\right] \\ &+ \sum_{i=1}^{N} Q_i^L(t)\mathbb{E}\left[\left(\left(1-x_i(t)\right)d_i(t) - \frac{f_i^l(t)\sigma}{c_i}\right)|\Theta(t)\right] \\ &+ \sum_{i=1}^{N} Q_i^E(t)\mathbb{E}\left[\left(B_i(t) - \frac{f_i(t)\sigma}{c_i}\right)|\Theta(t)\right] + V\cdot U(t).\end{aligned} \tag{37}$$

B. Problem Reformulation

It's evident that the drift-penalty function has an upper bound at each moment, represented by the right side of (37). Consequently, the original problem can be approximated by solving for the minimal value of this bound. The problem can be reformulated as follows:

$$\begin{aligned}\mathcal{P}_2: \min_{x,f,P}\ \mathbb{E}\Bigg\{ &B + \sum_{i=1}^{N} V_i(t)\mathbb{E}\left[\left(E_i(t)-e(t)\right)|\Theta(t)\right] \\ &+ \sum_{i=1}^{N} Q_i^L(t)\mathbb{E}\left[\left(\left(1-x_i(t)\right)d_i(t) - \frac{f_i^l(t)\sigma}{c_i}\right)|\Theta(t)\right] \\ &+ \sum_{i=1}^{N} Q_i^E(t)\mathbb{E}\left[\left(B_i(t) - \frac{f_i(t)\sigma}{c_i}\right)|\Theta(t)\right] + V\cdot U(t)\Bigg\}, \\ \text{s.t.}\quad &\text{Constraints}: (C_1)-(C_6).\end{aligned} \tag{38}$$

V. Lyapunov-Guided DRL Approach

A. MDP Modeling

The DRL method is used to solve the proposed problem. We first formulate the problem as a Markov decision process (MDP) to accurately describe the offloading and resource allocation decision processes. Here we present the elements of the MDP, including the state space, action space, and reward function.

- *State:* Let the state space of the vehicle i at time t be $s_i(t)$. The state space includes information like speed $v_k(t)$. Let $l_i(t)$ denote the longitude and latitude coordinates of the vehicle at time t. $f_i^{local\_max}(t)$ and f^{max} are separately the maximum computation resources at vehicle i and RSU. Therefore, the state space at time t can be expressed as:

$$S(t) = (s_1(t), s_2(t), \ldots, s_N(t)), \tag{39}$$

  where $s_i(t) = [l_i(t), v_k(t), f_i^{local\_max}(t), f^{max}, p_i^{max}, V_i(t), Q_i^L(t), Q_i^E(t)]$.
- *Action:* The action space consists of four parts: the computation offloading strategy $x_i(t)$, the edge computation resource allocation strategy $f_i(t)$ and $f^e(t)$, the local computation resource allocation strategy $f_i^l(t)$, and the transmission power resource allocation strategy $p_i(t)$. Therefore, the action space is given as,

$$A(t) = (a_1(t), \ldots, a_N(t), a_e(t)), \tag{40}$$

  where $a_i(t) = [x_i(t), f_i(t), f_i^l(t), p_i(t)]$ and $a_e(t) = f^e(t)$.
- *Reward:* The reward function of our MDP is defined as the negative of the Lyapunov optimization-based transformed objective of $\mathcal{P}_2$.

$$\begin{aligned}R(t) = -\Bigg\{ &B + \sum_{i=1}^{N} V_i(t)\mathbb{E}\left[\left(E_i(t)-e(t)\right)|\Theta(t)\right] \\ &+ \sum_{i=1}^{N} Q_i^L(t)\mathbb{E}\left[\left(\left(1-x_i(t)\right)d_i(t) - \frac{f_i^l(t)\sigma}{c_i}\right)|\Theta(t)\right] \\ &+ \sum_{i=1}^{N} Q_i^E(t)\mathbb{E}\left[\left(B_i(t) - \frac{f_i(t)\sigma}{c_i}\right)|\Theta(t)\right] \\ &+ V\cdot U(t)\Bigg\}.\end{aligned} \tag{41}$$

Algorithm 1: SAC-Based Algorithm for Computation Offloading and Resource Allocation.

Require: Initialize the parameters for the critic networks ω_1, ω_2, for the actor network π_θ, and for the target networks $\phi, \hat{\phi}$. Initialize replay memory D, the initial state s, and the parameter τ for soft updates.

Ensure: Optimal sequence of actions for computation offloading and resource allocation strategies.

```
Initialize parameters: ω1, ω2, φ, θ1, θ2, φ̂, πθ; empty replay memory D; set τ;
for episode e = 1, 2, ..., G do
    Initialize the simulation environment;
    Randomly generate and receive an initial state s;
    for t = 1, ..., T do
        Select action a_t according to the current policy πθ based on the state s;
        Execute action a_t in the environment;
        Observe reward r_t and new state s_{t+1};
        Store the transition (s_t, a_t, r_t, s_{t+1}) in the replay memory D;
        Sample a minibatch of K transitions (s_i, a_i, r_i, s_{i+1}) from D;
        for i = 1, ..., N do
            Calculate target value y_i = r_i + γ min_{j=1,2} Q_{ω_j}(s_{i+1}, π_θ(s_{i+1})) − ψ log π_θ(a_{i+1}|s_{i+1});
            Update critic networks by minimizing the loss: L(ω_j) = (1/N) Σ_i (Q_{ω_j}(s_i, a_i) − y_i)^2;
            Update actor network using the sampled policy gradient;
            Update target networks with soft update: ω1 ← τω1' + (1 − τ)ω1; ω2 ← τω2' + (1 − τ)ω2
        end for
    end for
end for
```

B. SAC-Based DRL Algorithm

Unlike traditional reinforcement learning algorithms that focus solely on cumulative reward maximization, SAC incorporates an entropy term to encourage exploration, thereby striking a balance between exploitation and exploration [37]. This entropy-augmented objective function leads to the development of a more robust and explorative policy compared to DDPG and other algorithms. Entropy is a measure of unpredictability or uncertainty of a system. If p is its probability density function, then the mathematical definition of entropy $H(X)$ for a discrete random variable X is given by:

$$H(X) = \mathbb{E}_{x\sim p}[-\log p(x)]. \tag{42}$$

In our algorithm, we aim to find a policy that maximizes the cumulative reward while remaining sufficiently unpredictable, hence improving exploration. The optimal policy in maximum entropy reinforcement learning is defined as:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi}\left[\sum_t r(s_t, a_t) + \psi H(\pi(\cdot|s_t))\right], \tag{43}$$

where ψ represents the temperature parameter which adjusts the relative importance of the entropy term against the reward.

This approach effectively balances the exploration-exploitation trade-off by not only encouraging the policy to seek higher rewards but also maintaining diversity in its action distribution. As illustrated in Fig. 3, the SAC-based algorithm utilizes five networks in total, which include one policy network, two value networks, and two target value networks. During the training process for each episode, the actor network selects an action a_t according to the current policy $\pi_\theta(s_t)$. This action is executed in the environment, leading to the next state s_{t+1} and receiving a reward r_t. The tuple (s_t, a_t, r_t, s_{t+1}) is stored in the replay buffer R.

For each learning step, a batch of transitions (s_i, a_i, r_i, s_{i+1}) is sampled from the replay buffer. The target value y_i for the Q-function is computed as the sum of the reward r_i and the discounted minimum Q-value of the next state and action, minus the log probability of the action, encouraging exploration. This is used to update the critic networks by minimizing the loss function, which is averaged over the batch:

$$L_Q(\omega_j) = \frac{1}{N}\sum_{i=1}^{N}\left(Q_{\omega_j}(s_i, a_i) - y_i\right)^2, \tag{44}$$

where $y_i = r_i + \gamma \min_{j=1,2} Q_{\omega_j}(s_{i+1}, \tilde{a}_{i+1}) - \psi \log \pi_\theta(\tilde{a}_{i+1}|s_{i+1})$ and $\tilde{a}_{i+1} \sim \pi_\theta(\cdot|s_{i+1})$.

The policy π is updated by minimizing the actor loss:

$$L_\pi(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\psi \log \pi_\theta(a_i|s_i) - \min_{j=1,2} Q_{\omega_j}(s_i, a_i)\right). \tag{45}$$

Finally, the parameters of the critic networks ω_1 and ω_2 are updated using soft updates with the target networks' parameters ω_1' and ω_2':

$$\omega_1^{updated} = \tau\omega_1' + (1-\tau)\omega_1, \tag{46}$$

$$\omega_2^{updated} = \tau\omega_2' + (1-\tau)\omega_2, \tag{47}$$

where τ represents the rate of the soft update.

VI. Performance Evaluation

A. Parameters Setting

All the simulated experiments are performed on a Python 3.11 platform with NVIDIA GeForce 2.1 GHz GPU, Intel(R) Xeon(R) Silver 4214R 2.40 GHz CPU and 32 GB of RAM. The related parameters of our system and algorithm are shown in Table III.

Assuming the RSU is elevated at 4 meters, positioned 1 m above the road surface and equipped with a range of 500 meters [38], it oversees a three-lane highway where each lane is 3.5 meters wide, according to the conventional road standards. And vehicles travel at a speed of 60km/h, and the system operates

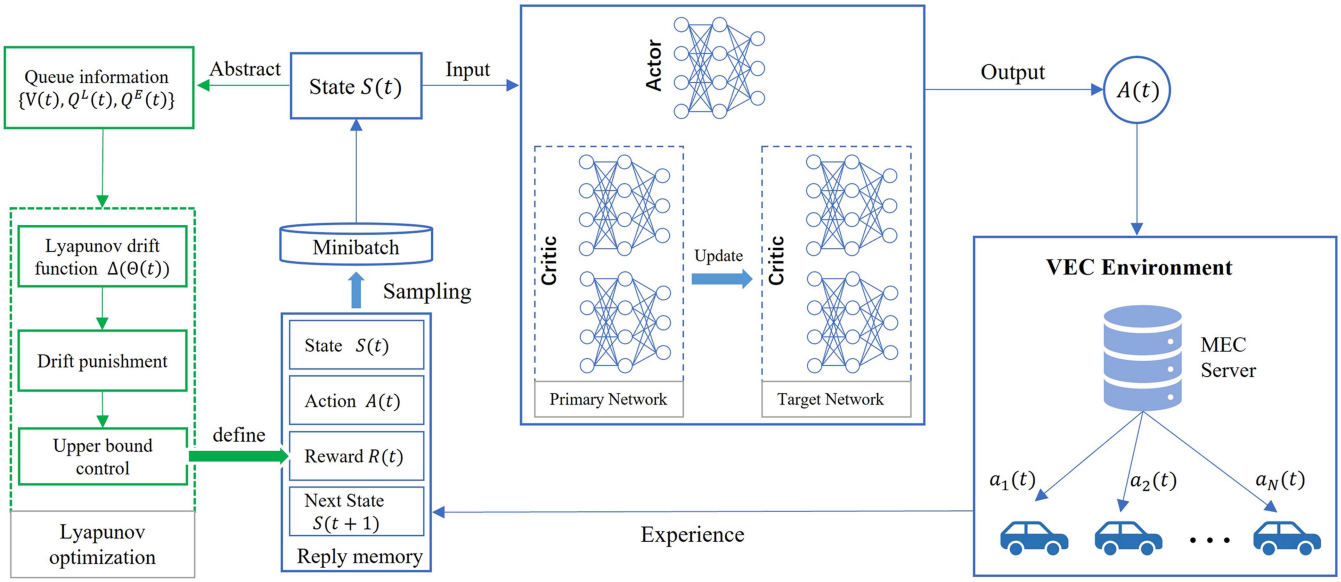

Fig. 3. The schematics of the proposed LySAC algorithm.

TABLE III
SIMULATION PARAMETERS

Parameter	Value
The number of vehicles $\mathcal{V}$	25
The duration of time slot σ	1 s
Number of episodes	1000
Task data size $d_i(t)$	0-2 Mbits
Coverage of RSUs	500 m
Vehicles speed	60 Km/h
Vehicle computation resource $f^{local_{max}}$	3 GHz
MEC server computation resource f^{max}	6 GHz
The maximum transmission power P_i^{max}	1 W
Number of CPU cycles processing one bit in vehicles c_i	3000-4500
Number of CPU cycles processing one bit in RSUs c_i	300
Energy coefficient κ	10^{-26}
Path loss exponent l	2.5
Gaussian white noise N_0	10^{-13}
Channel bandwidth W	20 MHz
Queue delay bound γ	500 ms
Communication gains CG	1.5
The control parameter V	5

with a slot duration of one second. The configuration of other parameters is primarily derived from [26] and [39]. We set the Channel bandwidth $W = 20$ MHz and energy coefficient $\kappa = 10^{-26}$. We assume the path loss exponent $l = 2.5$ and Gaussian white noise $N_0 = 10^{-13}$. Besides, the communication gains $CG = 1.5$ and the control parameter $V = 5$.

B. Numerical Normalization

In comparison to delay, energy consumption exhibits a significantly larger magnitude, leading to the two not being on the same scale. This discrepancy results in the minimal contribution of delay in the experimental results, thereby marginally influencing the offloading decisions for tasks.

To address this misalignment, normalization of both parameters becomes necessary. However, conventional normalization methods are deemed unsuitable due to the presence of unknown variables such as computational capacity and transmission power in the calculation of delay and energy consumption. In light of these challenges, the approach proposed in [40] is employed in this study, which can harmonize the magnitudes of delay and energy consumption to the same level. This is achieved by introducing a coefficient β such that $\frac{T}{\beta E} \approx 1$.

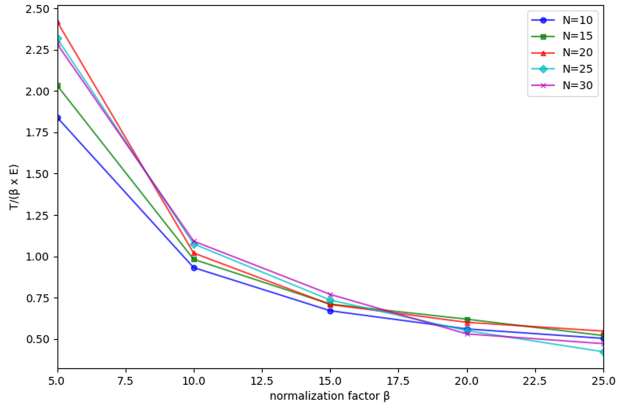

Fig. 4. Effect of different α on $T/(\beta \times E)$.

Experiments were conducted by varying the value of α under different numbers of vehicles, as illustrated in Fig. 4. As β increases, the value of $\frac{T}{\beta E}$ decreases. Notably, when $\beta = 10$, it can be observed that $\frac{T}{\beta E} \approx 1$ for various numbers of vehicles. Consequently, setting β to 10 ensures the scales of delay and energy consumption are unified for subsequent simulation experiments.

C. Baselines

In this study, we compare the proposed LySAC scheme with the following four algorithms:

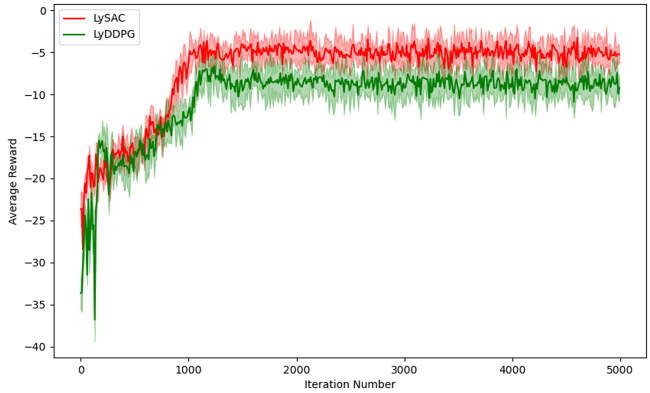

Fig. 5. The convergence of proposed schemes.

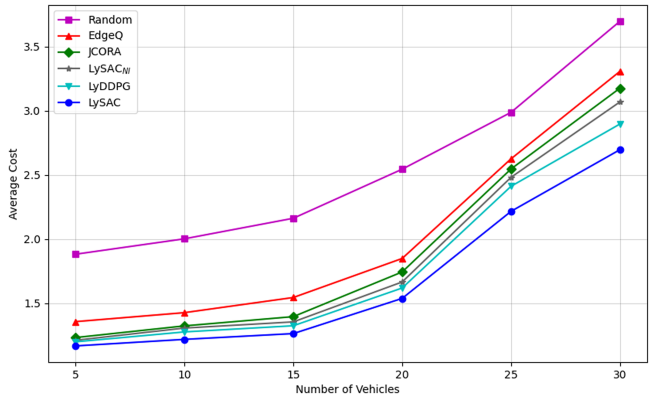

Fig. 6. System average cost varies with the number of vehicles.

- *Random:* This approach involves randomly deciding the offloading decisions, distributing computing resources evenly at the edge, utilizing maximum computing power locally, and setting communication power to the average value, specifically 0.5 watts.
- *EdgeQ:* The Edge and Queue-based Computation Resource Allocation (EdgeQ) strategy involves fully offloading tasks for edge processing, transmitting them via maximum communication power. At the edge, computation resources are allocated based on the length of the task queue for each vehicle.
- *JCORA:* The Joint Computation Offloading and Resource Allocation scheme [15] employs a two-step process. Initially, computational tasks are classified using the Advanced K-Means Task Classification to assess offloading feasibility. Subsequently, resource allocation is carried out through a DDQN-based algorithm.
- *LySAC$_{NI}$:* The Non-ISAC-aided LySAC algorithm, which means without considering the enhancement effect of ISAC technology on the system's communication rate, is otherwise identical to the LySAC.
- *LyDDPG:* In the proposed computation offloading and resource allocation scheme, the SAC algorithm is straightforwardly substituted with the DDPG algorithm for experimental comparison. This modified algorithm is named LyDDPG.

D. Convergence of Proposed Schemes

The graph Fig. 5 indicates that the average rewards for both the LySAC and LyDDPG algorithms improve as the number of episodes increases, which suggests that both algorithms are capable of learning and adapting their policies effectively across different learning rates. The variability in performance, as denoted by the shaded areas, can be attributed to the differences in the learning rates applied in each trial. And We adapt three distinct learning rates: 0.02, 0.003, and 0.005.

It is notable that the LySAC algorithm consistently outperforms the LyDDPG across all three learning rates, as evidenced by the higher average reward. Furthermore, the shaded areas suggest that the variability in the LySAC's performance is relatively small, particularly at higher episode counts, which could indicate that LySAC is less sensitive to changes in learning rate compared to LyDDPG. This could be interpreted as LySAC having a better capacity to find a more optimal policy across a range of learning rates, hence showing a more robust performance.

E. Average System Cost

The change of the average system cost is shown in Fig. 6. The cost is calculated as the weighted sum of the delay and ten times the energy consumption, with a lower value indicating better performance.

All algorithms experience an increase in the average cost as the number of vehicles grows. This trend can be attributed to the increased computational demand and network congestion, leading to higher delays and energy consumption for local computation and task transmission. The Random strategy, as in previous analyses, shows the steepest increase in cost with the addition of vehicles, which reflects its non-strategic and inefficient approach to task coordination. Its high cost is likely due to both significant delays, as previously observed, and potentially high energy consumption resulting from the lack of optimization in task allocation and processing. The EdgeQ algorithm exhibits a lower cost compared to the Random strategy, consistent with earlier observations where the EdgeQ method outperformed Random in terms of delay and the cost increase for EdgeQ is solely due to the energy used in transmitting tasks to the edge servers. For the deep learning-based algorithms–JCORA, LySAC$_{NI}$, LyDDPG and LySAC–the graph shows that they perform better than the non-deep learning algorithms in terms of average cost. Initially, when the number of vehicles is low, these algorithms maintain a low average cost, thanks to their efficient decision-making which minimizes both delay and energy consumption. However, as the number of vehicles exceeds 20, the average cost begins to rise more sharply. This could be due to the increased frequency of local computations and the associated energy consumption, as well as the higher transmission energy costs due to network congestion. Moreover, LySAC consistently shows the lowest average cost, suggesting that it is the most effective in balancing delay and energy consumption. Its performance indicates that LySAC can make more optimal decisions and showcases its potential for use in real-world vehicular networks.

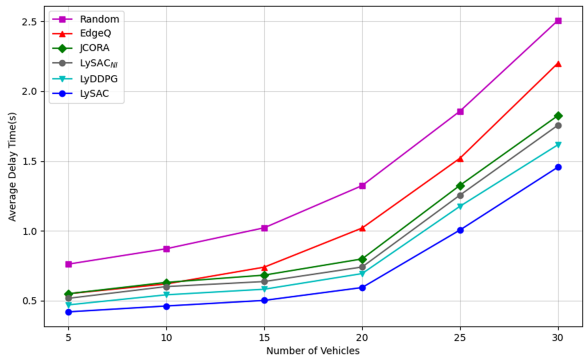

Fig. 7. System average delay varies with the number of vehicles.

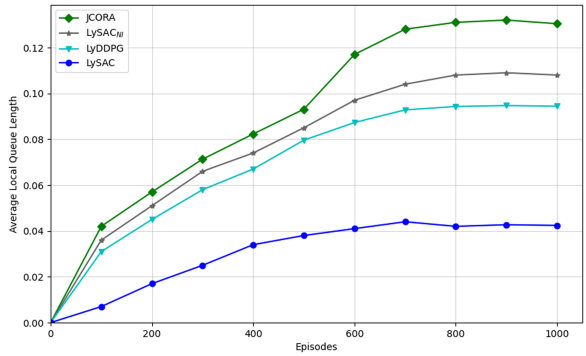

Fig. 8. Average local queue length with different algorithms.

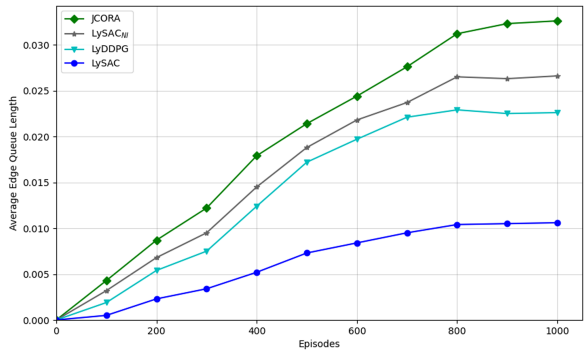

Fig. 9. Average edge queue length with different algorithms.

F. System Average Delay

Fig. 7 illustrates the system average delay time of five schemes. Notably, this comparison is conducted under the condition where the weight parameter α is set to 1, indicating the system is fully focused on delay without considering energy consumption. This specific setting underscores the prioritization of minimizing delay, providing a distinct perspective on the efficacy of each scheme under delay-sensitive conditions.

It is observable that the average delay increases as the number of vehicles grows, due to the limited computational resources available at the edge server. As the number of vehicles increases, the computational resources allocated to each vehicle decrease, resulting in an increase in latency. The Random strategy, which serves as a fundamental baseline among the evaluated algorithms, consistently exhibits the highest latency, highlighting the inefficacy of coordination schemes predicated on chance. In contrast, the EdgeQ algorithm, which offloads tasks entirely to the edge for execution, surpasses the random approach in performance. This improvement is due to the greater abundance of computational resources available at the edge compared to those on the vehicles themselves.

When the vehicle number within the system is relatively small, as at 5, 10, and 15, the increment in delay time is not pronounced, due to the edge servers' capacity to allocate ample computational resources per vehicle. However, as the vehicle count exceeds 20, the edge computational power available per vehicle dwindles, and the system's task queue begins to congest, precipitating a rapid acceleration in the growth rate of the average delay. Notably, the three deep learning-based algorithms outperform the first two, demonstrating their superior adaptability to dynamic environments and complex networks. Among these, the LyDDPG algorithm performs superiorly relative to JCORA, which may be attributed to the latter's reliance on an advanced K-means clustering for offloading decisions that do not guarantee optimal decision-making as comprehensively as LyDDPG and LySAC. The LySAC scheme, proposed in this study, consistently maintains the lowest average delay across the evaluated algorithms, demonstrating its potent adaptability to the highly dynamic environment of vehicular networks and its efficacy in managing computational and communicative resources efficiently.

G. The Queue Stability Performance

The two graphs, Figs. 8 and 9 illustrate the convergence behavior of average queue lengths for local and edge queues, respectively. However, it should be noted that the comparison of all algorithms may not be necessary, as the Random and Edge solutions do not incorporate control queue technology. Therefore, our comparison is focused solely on the four algorithms that feature queue control technology: JCORA, LyDDPG, LySAC$_{NI}$ and LySAC.

First, for the local queue length (Fig. 8), it's evident that the LySAC algorithm achieves a lower queue length compared to the other three schemes as the number of episodes increases. LySAC's performance stabilizes quickly, maintaining a relatively flat curve after an initial growth phase. This suggests that LySAC efficiently manages local computational tasks, preventing backlog buildup and thus indicating an effective training process that likely includes Lyapunov optimization principles for stability and optimality.

Second, the edge queue length (Fig. 9) follows a similar pattern, with LySAC demonstrating superior performance over the other two algorithms. It shows a steady rise initially, but then plateaus, indicating that it reaches a stable queue length swiftly. This plateauing implies that LySAC is effectively balancing the offloading process to the edge, avoiding excessive queue length buildup, which is critical for maintaining system performance.

In comparison, both LyDDPG and JCORA exhibit a more pronounced increase in queue lengths for both local and edge queues. JCORA shows a steady but more gradual climb than

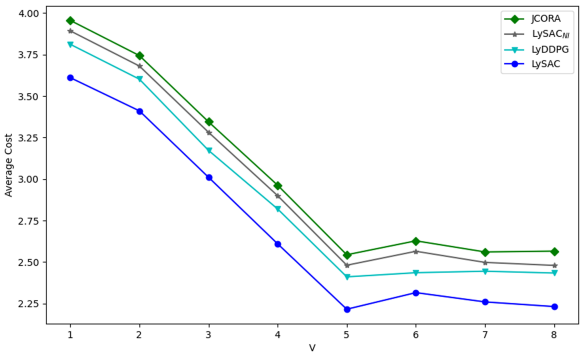

Fig. 10. System average cost versus the control parameter V.

LySAC, suggesting a slower convergence towards an optimal policy. LyDDPG, while outperforming LCORA at the edge, still falls behind LySAC in the performance of queue length control, indicating that while it manages edge offloading adequately, it does not optimize as efficiently as LySAC. In summary, LySAC not only achieves lower queue lengths at both local and edge levels but also demonstrates quicker stabilization, reflecting its effective learning and decision-making processes. This highlights LySAC's potential to reach optimal performance more rapidly and maintain system stability, making it a preferable choice for managing vehicular network tasks.

H. Impact of Control Parameter V

Fig. 10 presents the relationship between the system average cost and the control parameter V, which serves as a balancing coefficient, moderating the trade-off between cost factors and queue length within the system. The average cost initially decreases as V increases, then subsequently stabilizes, indicating a complex trade-off. At lower values, a focus on cost reduction seems effective, but as V grows, this results in longer queues. Such an increase in queue lengths may, in turn, exacerbate delay and energy consumption, consequently elevating the overall system cost at higher values of V. This trend highlights the importance of maintaining a balanced V to optimize the overall performance of the system.

Fig. 10 shows that choosing a common optimal control parameter V is crucial for the four algorithms. When V is set to 5, the empirical data suggest that all four algorithms achieve their minimum system average cost. With the right V, it is possible to fine-tune the system for enhanced cost efficiency while preserving the stability of queue lengths, thereby ensuring a balanced operational framework.

VII. Conclusion and Future Work

In this paper, we propose a joint computation offloading decision and resource allocation strategy in the VEC network, designed to ensure minimal latency and energy consumption. To tackle this challenge, we adopt the weighted sum of energy consumption and delay as our objective function while ensuring queue stability. Initially, we employ Lyapunov optimization to transform the problem, securing task queue stability without prior information. The transformed optimization problem, characterized by its non-convexity and constraints on transmission power, radio resources, and edge server capacity, proves challenging for traditional methods. Therefore, we introduce an algorithm based on SAC reinforcement learning for joint decision-making. Our simulation experiments demonstrate the stability and effectiveness of our proposed method. Compared to existing reinforcement learning approaches, it significantly reduces latency and energy consumption while maintaining queue length stability. However, while our method performs well in simulations, it may encounter challenges in practical implementation due to its complexity and the impact of environmental factors. Moreover, our study relies primarily on simulation data and lacks validation through real-world experiments, which somewhat limits the applicability of our method. Yet, these drawbacks are more or less unavoidable in related literature. Additionally, our decision-making process is semi-centralized and executed on edge servers, which could hinder practicality in dynamic and complex vehicular network environments. In future research, we intend to explore a hybrid of distributed and centralized decision-making strategies, such as employing multi-agent reinforcement learning. This approach would allow individual vehicles to autonomously select offloading options, while the edge server would play a partial decision-making and supportive role in the process.

References

[1] M. Noor-A-Rahim et al., "6G for vehicle-to-everything (V2X) communications: Enabling technologies, challenges, and opportunities," *Proc. IEEE*, vol. 110, no. 6, pp. 712–734, Jun. 2022.
[2] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular edge computing and networking: A survey," *Mobile Netw. Appl.*, vol. 26, pp. 1145–1168, 2021.
[3] C. Li, L. Chai, K. Jiang, Y. Zhang, J. Liu, and S. Wan, "DNN partition and offloading strategy with improved particle swarm genetic algorithm in VEC," *IEEE Trans. Intell. Veh.*, to be published, doi: 10.1109/TIV.2023.3346506.
[4] P. Plotnikov, G. Tambovtsev, and A. Vladyko, "Performance evaluation of V2X model with a mobile road side units," in *Proc. Intell. Technol. Electron. Devices Veh. Road Transport Complex*, 2023, pp. 1–4.
[5] R. Meneguette, R. De Grande, J. Ueyama, G. P. R. Filho, and E. Madeira, "Vehicular edge computing: Architecture, resource management, security, and challenges," *ACM Comput. Surv.*, vol. 55, no. 1, pp. 1–46, 2021.
[6] C. Yang, B. Liu, H. Li, B. Li, K. Xie, and S. Xie, "Learning based channel allocation and task offloading in temporary UAV-assisted vehicular edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 71, no. 9, pp. 9884–9895, Sep. 2022.
[7] X. Zhang, M. Peng, S. Yan, and Y. Sun, "Joint communication and computation resource allocation in fog-based vehicular networks," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 13195–13208, Aug. 2022.
[8] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019.
[9] A. Liu et al., "A survey on fundamental limits of integrated sensing and communication," *IEEE Commun. Surv. Tut.*, vol. 24, no. 2, pp. 994–1034, Second Quarter 2022.
[10] Z. Wei, F. Liu, C. Masouros, N. Su, and A. P. Petropulu, "Toward multifunctional 6G wireless networks: Integrating sensing, communication, and security," *IEEE Commun. Mag.*, vol. 60, no. 4, pp. 65–71, Apr. 2022.
[11] F. Liu et al., "Integrated sensing and communications: Toward dual-functional wireless networks for 6G and beyond," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 6, pp. 1728–1767, Jun. 2022.
[12] Y. Cui, F. Liu, C. Masouros, J. Xu, T. X. Han, and Y. C. Eldar, "Integrated sensing and communications: Background and applications," in *Integrated Sensing and Communications*. Berlin, Germany: Springer, 2023, pp. 3–21.

[13] X. Cheng, D. Duan, S. Gao, and L. Yang, "Integrated sensing and communications (ISAC) for vehicular communication networks (VCN)," *IEEE Internet Things J.*, vol. 9, no. 23, pp. 23441–23451, Dec. 2022.
[14] J. Mu, Y. Gong, F. Zhang, Y. Cui, F. Zheng, and X. Jing, "Integrated sensing and communication-enabled predictive beamforming with deep learning in vehicular networks," *IEEE Commun. Lett.*, vol. 25, no. 10, pp. 3301–3304, Oct. 2021.
[15] Q. Liu, R. Luo, H. Liang, and Q. Liu, "Energy-efficient joint computation offloading and resource allocation strategy for ISAC-aided 6G V2X networks," *IEEE Trans. Green Commun. Netw.*, vol. 7, no. 1, pp. 413–423, Mar. 2023.
[16] Y. Li, F. Liu, Z. Du, W. Yuan, and C. Masouros, "ISAC-Enabled V2I networks based on 5G NR: How much can the overhead be reduced?," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2023, pp. 691–696.
[17] Z. Bai, F. Hou, and H. Shan, "Optimal sensing time design in ISAC-enabled vehicular networks," in *Proc. IEEE/CIC Int. Conf. Commun. China*, 2023, pp. 1–6.
[18] S. Li, J. Chen, X. Kuai, and Y.-C. Liang, "AmBC-aided integrated sensing and communication systems for V2X networks," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2023, pp. 1480–1485.
[19] N. Huang et al., "Mobile edge computing aided integrated sensing and communication with short-packet transmissions," *IEEE Trans. Wireless Commun.*, vol. 23, no. 7, pp. 7759–7774, Jul. 2024.
[20] H. Tang, H. Wu, G. Qu, and R. Li, "Double deep Q-network based dynamic framing offloading in vehicular edge computing," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 3, pp. 1297–1310, May/Jun. 2023.
[21] X. Huang, L. He, X. Chen, L. Wang, and F. Li, "Revenue and energy efficiency-driven delay-constrained computing task offloading and resource allocation in a vehicular edge computing network: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 8852–8868, Jun. 2021.
[22] A. S. Kumar, L. Zhao, and X. Fernando, "Task offloading and resource allocation in vehicular networks: A Lyapunov-based deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 72, no. 10, pp. 13360–13373, Oct. 2023.
[23] G. Qu, H. Wu, R. Li, and P. Jiao, "DMRO: A deep meta reinforcement learning-based task offloading framework for edge-cloud computing," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 3, pp. 3448–3459, Sep. 2021.
[24] S.-C. Hung, X. Zhang, A. Festag, K.-C. Chen, and G. Fettweis, "Vehicle-centric network association in heterogeneous vehicle-to-vehicle networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 5981–5996, Jun. 2019.
[25] A. Asheralieva and D. Niyato, "Game theory and Lyapunov optimization for cloud-based content delivery networks with device-to-device and UAV-enabled caching," *IEEE Trans. Veh. Technol.*, vol. 68, no. 10, pp. 10094–10110, Oct. 2019.
[26] S. Bi, L. Huang, H. Wang, and Y.-J. A. Zhang, "Lyapunov-guided deep reinforcement learning for stable online computation offloading in mobile-edge computing networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 11, pp. 7519–7537, Nov. 2021.
[27] S. Li, G. Zhu, and S. Lin, "Joint radio and computation resource allocation with predictable channel in vehicular edge computing," in *Proc. 21st Int. Conf. Intell. Transp. Syst.*, 2018, pp. 3736–3741.
[28] A. Mekrache, A. Bradai, E. Moulay, and S. Dawaliby, "Deep reinforcement learning techniques for vehicular networks: Recent advances and future trends towards 6G," *Veh. Commun.*, vol. 33, 2022, Art. no. 100398.
[29] P. Ladosz, L. Weng, M. Kim, and H. Oh, "Exploration in deep reinforcement learning: A survey," *Inf. Fusion*, vol. 85, pp. 1–22, 2022.
[30] H. Hua, Y. Li, T. Wang, N. Dong, W. Li, and J. Cao, "Edge computing with artificial intelligence: A machine learning perspective," *ACM Comput. Surv.*, vol. 55, no. 9, pp. 1–35, 2023.
[31] M. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. Berlin, Germany: Springer, 2022.
[32] C. Pukdeboon, "A review of fundamentals of Lyapunov theory," *J. Appl. Sci*, vol. 10, no. 2, pp. 55–61, 2011.
[33] P. Zhou, X. Hu, Z. Zhu, and J. Ma, "What is the most suitable Lyapunov function?," *Chaos, Solitons Fractals*, vol. 150, 2021, Art. no. 111154.
[34] K. Sadatdiynov, L. Cui, L. Zhang, J. Z. Huang, S. Salloum, and M. S. Mahmud, "A review of optimization methods for computation offloading in edge computing networks," *Digit. Commun. Netw.*, vol. 9, no. 2, pp. 450–461, 2023.
[35] S. Lu, X. Meng, Z. Du, Y. Xiong, and F. Liu, "On the performance gain of integrated sensing and communications: A subspace correlation perspective," in *Proc. IEEE Int. Conf. Commun.*, 2023, pp. 2735–2740.
[36] J. Wang, N. Varshney, C. Gentile, S. Blandino, J. Chuang, and N. Golmie, "Integrated sensing and communication: Enabling techniques, applications, tools and data sets, standardization, and future directions," *IEEE Internet Things J.*, vol. 9, no. 23, pp. 23416–23440, Dec. 2022.
[37] T. Haarnoja et al., "Soft actor-critic algorithms and applications," 2018, *arXiv: 1812.05905*.
[38] C. Creß, Z. Bing, and A. C. Knoll, "Intelligent transportation systems using roadside infrastructure: A literature survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 7, pp. 6309–6327, Jul. 2024.
[39] S. Huang, M. Zhang, Y. Gao, and Z. Feng, "MIMO radar aided mmwave time-varying channel estimation in MU-MIMO V2X communications," *IEEE Trans. Wireless Commun.*, vol. 20, no. 11, pp. 7581–7594, Nov. 2021.
[40] J. Zhang et al., "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2633–2645, Aug. 2018.

Yonghui Liang received the BSc degree from the Jiangxi University of Finance and Economics, China, in 2020. He is currently working toward the MS degree with the Center for Applied Mathematics, Tianjin University, China. His research interests include Internet of Things, mobile edge computing, and deep learning.

Huijun Tang (Member, IEEE) received the BSc degree from Jinan University, China, in 2016 and the MS and PhD degree from Tianjin University, China, in 2018 and 2022, respectively. She is currently a lecturer with the School of Cyberspace, Hangzhou Dianzi University. Her research interests include Internet of Things, mobile edge computing, and deep learning.

Huaming Wu (Senior Member, IEEE) received the BE and MS degrees from the Harbin Institute of Technology, China, in 2009 and 2011, respectively, both in electrical engineering, and the PhD degree in the highest honor in computer science from Freie Universität Berlin, Germany, in 2015. He is currently a professor at the Center for Applied Mathematics, Tianjin University, China. His research interests include mobile cloud computing, edge computing, Internet of Things, deep learning, complex networks, and DNA storage.

Yixiao Wang received the BSc degree from Minzu University of China, in 2021, the MS degree from the Center for Applied Mathematics, Tianjin University, China, in 2024. His research interests include Internet of Things, reinforcement learning, and deep optimization.

Pengfei Jiao (Member, IEEE) received the PhD degrees in computer science from Tianjin University, Tianjin, China, in 2018. From 2018 to 2021, he was a lecturer with the Center of Biosafety Research and Strategy of Tianjin University. He is currently a professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His current research interests include complex network analysis and its applications.