Multi-Agent Reinforcement Learning for SLA-Aware Network Slicing in UAV-Enabled MEC

Mohammad Farhoudi[1], Zeinab Sasan[2], Masoud Shokrnezhad[3], and Tarik Taleb[4]
[1] *Oulu University, Oulu, Finland*; mohammad.farhoudi@oulu.fi
[2] *Amirkabir University of Technology, Tehran, Iran*; z.sasan@aut.ac.ir
[3] *ICTFICIAL Oy, Espoo, Finland*; masoud.shokrnezhad@ictficial.com
[4] *Ruhr University Bochum (RUB), Bochum, Germany*; tarik.taleb@rub.de

***Abstract*—Unmanned Aerial Vehicle (UAV)-enabled Mobile Edge Computing (MEC) offers flexible capacity provisioning for heterogeneous network slices, including Hyper-Reliable and Low-Latency Communication (HRLLC), Enhanced Mobile Broadband (eMBB), and Massive Machine-Type Communications (mMTC). However, guaranteeing slice-level Service-Level Agreements (SLAs) under dynamic user mobility, stochastic task arrivals, and constrained onboard energy and computing resources remains a fundamental challenge. This paper proposes a predictive multi-agent Reinforcement Learning (RL) framework that proactively maintains SLA stability in UAV-enabled MEC through coordinated trajectory control and computation resource allocation. A lightweight prediction module forecasts near-future user mobility, enabling UAVs to anticipate congestion and reposition before SLA violations occur. We design an SLA-aware reward function that explicitly penalizes both violation probability and duration across slices, alongside total energy consumption. UAV agents are trained using Multi-Agent Proximal Policy Optimization (MAPPO) with centralized training and decentralized execution, enabling scalable online decision-making. Event-driven simulations with realistic mobility traces demonstrate that the proposed framework significantly improves SLA stability compared with baselines while maintaining competitive energy efficiency and delay performance, approaching oracle-level performance with sufficiently accurate predictive information.**

***Index Terms*—UAV-enabled MEC, network slicing, SLA-aware resource allocation, multi-agent reinforcement learning, MAPPO.**

I. Introduction

The rapid proliferation of computation-intensive and delay-sensitive applications, such as augmented reality, autonomous systems, and real-time video analytics, has imposed stringent requirements on next-generation wireless networks [1]. Mobile Edge Computing (MEC) has emerged as a key enabler to address these challenges by bringing computational resources closer to end users, thereby reducing delay and alleviating backhaul congestion [2]. Meanwhile, Unmanned Aerial Vehicles (UAVs), due to their flexibility, rapid deployment, and communication capabilities, have been increasingly integrated into MEC systems to provide on-demand edge services in scenarios with limited or damaged infrastructure, such as remote monitoring and temporary hotspots. In parallel, network slicing by logically partitioning network resources into multiple isolated slices, enables customized service provisioning for applications with distinct performance requirements, such as Hyper Reliable and Low-Latency Communication (HRLLC), Enhanced Mobile Broadband (eMBB), and Massive Machine-Type Communication (mMTC) [3], [4]. The integration of UAV-enabled MEC with network slicing offers a promising paradigm for delivering flexible and efficient edge intelligence in dynamic environments.

However, realizing this vision introduces significant technical challenges. In UAV-enabled MEC systems with network slicing, multiple UAVs should serve ground users with heterogeneous slice requirements while jointly optimizing their trajectory planning, user association, and computation resource allocation. Each slice imposes distinct Service-Level Agreement (SLA) constraints on tolerable delay, requiring careful coordination between communication and computation resources. The problem is further complicated by UAV mobility constraints, limited onboard energy budgets, and constrained computation capacity. Moreover, the system should operate under dynamic and uncertain conditions, including time-varying user mobility, stochastic task arrivals with different characteristics, and evolving channel conditions. These factors result in a complex, stochastic, and time-coupled optimization problem where current UAV positions and energy states influence future system dynamics.

Extensive research has investigated UAV-enabled MEC systems from multiple perspectives. Several works applied deep Reinforcement Learning (RL) approaches to optimize UAV trajectory and resource orchestration in UAV-enabled systems, focusing primarily on system efficiency and energy consumption [5], [6]. In parallel, some studies introduced network slicing concepts into UAV-MEC systems. For instance, [7]–[9] proposed slicing frameworks that emphasize resource partitioning, survivability mechanisms, and architectural design for heterogeneous service provisioning. Another research direction addressed task offloading optimization, where Tian *et al.* [10] and Chen *et al.* [11] developed user satisfaction and Quality of Service (QoS)-oriented offloading schemes in multi-UAV settings. Also, Li *et al.* [12] introduced a self-adjusting network slicing mechanism using two-timescale RL, which adapts slice configurations based on network dynamics, representing an advancement toward integrating slicing and learning-based control in UAV-MEC systems.

Despite these advancements, existing works mainly optimize conventional QoS metrics such as delay or throughput.

TABLE I
COMPARISON OF EXISTING WORKS AND THE PROPOSED METHOD

Ref.	Main Focus	UAV-MEC	Slicing	Trajectory	Learning	SLA-Aware	Predictive	Main Limitation
[5]	Service composition in aerial-terrestrial networks	✓	✗	✗	✓	✗	✓	No slicing and no SLA-aware control
[6]	Energy-efficient orchestration in 6G aerial-terrestrial	✓	✗	✓	✓	✗	✓	Focus on energy and QoS, no slicing
[7]	Survivable resource slicing in UAV-MEC	✓	✓	✗	✓	✗	✓	No trajectory/offloading joint optimization
[8]	SDN-based slicing architecture for UAV-MEC	✓	✓	✗	✗	✗	✗	Mostly architectural, no dynamic control
[9]	5G slice extension with UAV-MEC	✓	✓	✗	✓	✗	✗	Only slice extension, no multi-slice orchestration
[10]	User satisfaction-based task offloading	✓	✗	✓	✗	✗	✗	No slicing and no SLA guarantees
[11]	QoS-aware task offloading in multi-UAV MEC	✓	✗	✓	✓	✗	✗	QoS-based, no SLA modeling
[12]	Dynamic self-adjusting network slicing	✓	✓	✓	✓	✗	✓	No explicit SLA violation modeling
This work	**SLA-stable slicing with predictive multi-agent learning**	✓	✓	✓	✓	✓	✓	**SLA-aware slicing, trajectory, and offloading**

However, such metrics are insufficient for guaranteeing SLAs, which require strict and often probabilistic guarantees on performance metrics. As shown in Table I, most existing approaches treat slicing, trajectory control, and resource orchestration as separate problems, or rely on reactive mechanisms that adapt only after performance degradation occurs. In practical multi-service environments, different slices have heterogeneous and time-varying requirements, and maintaining stable SLA satisfaction under user mobility, stochastic traffic arrivals, and UAV energy constraints remains a critical challenge. As a result, the fundamental problem of *SLA stability in UAV-enabled MEC network slicing* remains largely unexplored.

To address these challenges, this paper proposes a novel framework for *SLA-aware network slicing in UAV-enabled MEC systems*, where UAVs serve as dynamic orchestrators for maintaining SLA guarantees across multiple service slices. The main paper's contributions are summarized as follows:

- We formulate a joint optimization problem that integrates UAV trajectory control, user association, and slice-level resource allocation to minimize energy consumption and SLA violation probability and duration across slices.
- We develop a predictive multi-agent RL framework based on Multi-Agent Proximal Policy Optimization (MAPPO), where each UAV leverages user mobility predictions to proactively prevent SLA violations.
- Simulations demonstrate that the proposed approach outperforms baseline methods in terms of SLA satisfaction, temporal stability, and energy efficiency.

In the rest: Section II presents the system model and problem formulation, Section III describes the proposed methodology, Section IV evaluates the performance through simulations, and Section V concludes the paper.

II. SYSTEM MODEL AND PROBLEM FORMULATION

We consider a UAV-enabled MEC system, where UAVs provide computation offloading services for ground users, as depicted in Fig. 1. The system operates over a finite time horizon $\mathcal{T}$, divided into discrete time slots indexed by t and with duration Δt. The set of ground users is denoted by $\mathcal{K} = \{1, \ldots, K\}$. Users generate computation-intensive tasks, which are processed within the same time slot in which they are generated, and offloaded to UAVs for remote execution. To capture dynamic task arrivals, we define a binary task-arrival indicator $\lambda_k(t) \in \{0, 1\}$, where $\lambda_k(t) = 1$ indicates that user k generates a computation task at time slot t. The set of UAVs

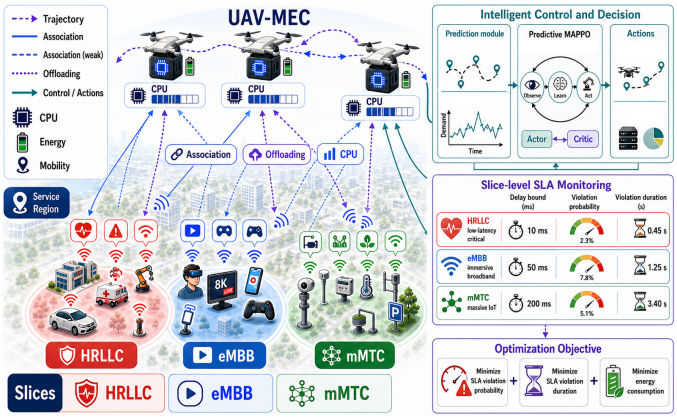

Fig. 1. System model, including ground users grouped into heterogeneous slices with slice-level SLA requirements. UAVs act as flying edge servers.

is denoted by $\mathcal{U} = \{1, \ldots, U\}$, where each is equipped with communication and computation capabilities, acting as flying edge servers. The maximum computation capacity $F_u^{\max}$, maximum speed $V_u^{\max}$, and energy budget $E_u^{\max}$ for each UAV u represent constrained onboard resources.

1) Network Slicing Model: The system supports multiple network slices to serve heterogeneous applications with diverse service requirements. The set of slices is denoted by $\mathcal{S} = \{1, \ldots, S\}$, where each slice s represents a logical service class such as HRLLC, eMBB, or mMTC, aligned with representative Sixth Generation (6G) usage scenarios [13]. The class of slice s specifies its offload task's profile, defined as

$$\xi_s = \{\bar{D}_s, \bar{C}_s, \bar{\tau}_s^{\max}\}, \tag{1}$$

where $\bar{D}_s$ represents the nominal input data size, $\bar{C}_s$ denotes the nominal required CPU cycles, and $\tau_s^{\max}$ is the maximum tolerable delay for slice s. We denote the set of active users associated with slice s by $\mathcal{K}_s(t)$, where $\mathcal{K}_s(t) \subseteq \mathcal{K}$.

The generated task of user k belongs to the slice s_k, whose characteristics are determined by the corresponding profile ξ_{s_k}. In particular, $\tau_k^{\max} = \bar{\tau}_{s_k}^{\max}$, while $D_k(t)$ and $C_k(t)$ are generated according to slice-dependent distributions around $\bar{D}_{s_k}$ and $\bar{C}_{s_k}$. The task is expressed by $\{D_k(t), C_k(t), \tau_k^{\max}, \mathbf{w}_k(t)\}$ with $\mathbf{w}_k(t) = [x_k(t), y_k(t), z_k(t)]$ represent the location of k at time slot t, where $x_k(t)$ and $y_k(t)$ refer to the horizontal coordinates, and $z_k(t) = 0$ denotes the altitude for ground users.

2) UAV Mobility Model: UAV u moves in a three-dimensional space with position $\mathbf{q}_u(t) = [x_u(t), y_u(t), z_u(t)]$ that directly affects the distance to users. Due to physical mo-

bility limitations, the displacement of each UAV between two consecutive time slots is constrained by $V_u^{\max}$. Accordingly, the mobility constraint of UAV u is expressed as

$$\|\mathbf{q}_u(t+1) - \mathbf{q}_u(t)\| \leq V_u^{\max}\Delta t, \quad \forall u \in \boldsymbol{\mathcal{U}},\, t \in \boldsymbol{\mathcal{T}}. \quad (2)$$

3) User Association Model: Active user k is associated with and offloads its task to UAV u, which is indicated by a binary variable $a_{k,u}(t) \in \{0,1\}$. Specifically, $a_{k,u}(t) = 1$ means that user k offloads its task to UAV u (0 otherwise). Each active user is assumed to be served by one UAV at each time slot; therefore, the association constraint is expressed as

$$\textstyle\sum_{u\in\boldsymbol{\mathcal{U}}} a_{k,u}(t) = \lambda_k(t), \quad \forall k \in \boldsymbol{\mathcal{K}}, t \in \boldsymbol{\mathcal{T}}. \quad (3)$$

We assume that users maintain connectivity with the selected UAV, where larger distances are reflected through reduced transmission rates.

4) Communication Model: The achievable transmission rate between user k and UAV u depends on (i) their relative distance $d_{k,u}(t) = \|\mathbf{q}_u(t) - \mathbf{w}_k(t)\|$, (ii) channel conditions, and (iii) transmit power. The channel gain is modeled using a distance-dependent path-loss exponent γ and channel gain $h_{k,u}(t) = \frac{\beta_0}{d_{k,u}^{\gamma}(t)}$ at a reference distance β_0. The transmit power is not treated explicitly as an optimization variable in practical modeling; rather, it is taken as a distance-aware power control mechanism, where users adapt their transmit power based on the communication distance. Specifically, the transmit power of user k when communicating with UAV u is modeled as

$$P_{k,u}(t) = P_0 \left(\frac{d_{k,u}(t)}{d_0}\right)^{\rho}, \quad (4)$$

where P_0 is the reference transmit power at distance d_0, and ρ controls the degree of path-loss compensation. In particular, $\rho < \gamma$ indicates partial compensation, in which the received signal quality degrades with distance. With channel bandwidth B and noise power σ^2, the transmission rate is given by

$$R_{k,u}(t) = B\log_2\left(1 + \frac{P_{k,u}(t)\, h_{k,u}(t)}{\sigma^2}\right). \quad (5)$$

Accordingly, the transmission delay required to upload the input data to the selected UAV is given by

$$T_k^{\mathrm{tx}}(t) = \textstyle\sum_{u\in\boldsymbol{\mathcal{U}}} a_{k,u}(t)\frac{D_k(t)}{R_{k,u}(t)}. \quad (6)$$

5) Computation Model: Each UAV allocates its CPU resource to its associated users. The amount of CPU cycles allocated by UAV u to user k at time slot t is denoted by $f_{k,u}(t)$, which should satisfy $f_{k,u}(t) \geq 0$. The allocated resources should be sufficient to complete the task within the considered time scale. The computation model is defined as

$$T_k^{\mathrm{comp}}(t) = \textstyle\sum_{u\in\boldsymbol{\mathcal{U}}} a_{k,u}(t)\frac{C_k(t)}{f_{k,u}(t)}. \quad (7)$$

To model the resource limitations, the total allocated resources $f_{k,u}(t)$ for all connected users cannot exceed UAV u's maximum capacity, expressed as

$$\textstyle\sum_{k\in\boldsymbol{\mathcal{K}}} a_{k,u}(t) f_{k,u}(t) \leq F_u^{\max}, \quad \forall u \in \boldsymbol{\mathcal{U}}, t \in \boldsymbol{\mathcal{T}}. \quad (8)$$

6) SLA Violation: The SLA violation manifests itself in three ways: (i) user-level SLA violation, (ii) slice-level SLA violation, and (iii) SLA violation duration. User-level violation occurs when the task completion delay exceeds the task's maximum tolerable delay, expressed as

$$I_k(t) = 1 \text{ if } \tau_k(t) = T_k^{\mathrm{tx}}(t) + T_k^{\mathrm{comp}}(t) > \tau_k^{\max}. \quad (9)$$

At each time slot t, the instantaneous SLA violation ratio of slice s is defined as

$$P_s^{\mathrm{viol}}(t) = \tfrac{1}{|\mathcal{K}_s(t)|}\textstyle\sum_{k\in\mathcal{K}_s(t)} I_k(t), \quad (10)$$

which denotes the fraction of users in $\mathcal{K}_s(t)$ whose SLAs are violated. We introduced the long-term SLA violation as $\bar{P}_s^{\mathrm{viol}} = \frac{1}{|\boldsymbol{\mathcal{T}}|}\sum_{t\in\boldsymbol{\mathcal{T}}} P_s^{\mathrm{viol}}(t)$ that captures the SLA violation experienced by s over time. Since SLA degradation may persist over multiple time slots, we also quantify its temporal persistence by defining the normalized SLA violation duration of slice s over the time horizon as

$$\tilde{T}_s^{\mathrm{viol}} = \tfrac{1}{|\boldsymbol{\mathcal{T}}|}\textstyle\sum_{t\in\boldsymbol{\mathcal{T}}} \mathbb{I}\left(P_s^{\mathrm{viol}}(t) > \epsilon_s\right), \quad (11)$$

where ϵ_s is a predefined violation threshold, and $\mathbb{I}(\cdot)$ denotes the indicator function.

7) UAV Energy Model: The total UAV energy consumption over the time horizon consists of (i) propulsion (flight) energy for movement and (ii) computation energy for task processing

$$E_u = \textstyle\sum_{t\in\boldsymbol{\mathcal{T}}}\left(E_u^{\mathrm{fly}}(t) + E_u^{\mathrm{comp}}(t)\right) \leq E_u^{\max},\ \forall u \in \boldsymbol{\mathcal{U}}, \quad (12)$$

that should not exceed the available energy budget. The propulsion energy is modeled as a tractable approximation of UAV displacement between consecutive time slots

$$E_u^{\mathrm{fly}}(t) = \varsigma\left\|\mathbf{q}_u(t+1) - \mathbf{q}_u(t)\right\|^2, \quad (13)$$

where ς is a propulsion-energy coefficient. With the energy consumption coefficient per CPU cycle η, the computation energy utilized at time slot t is modeled as

$$E_u^{\mathrm{comp}}(t) = \eta\textstyle\sum_{k\in\boldsymbol{\mathcal{K}}} a_{k,u}(t) f_{k,u}(t). \quad (14)$$

8) Problem Formulation: The optimization objective

$$\min_{\mathbf{q},a,f} \sum_{s\in\boldsymbol{\mathcal{S}}} b_s \bar{P}_s^{\mathrm{viol}} + \sum_{s\in\boldsymbol{\mathcal{S}}} \beta_s \tilde{T}_s^{\mathrm{viol}} + \chi \sum_{u\in\boldsymbol{\mathcal{U}}} E_u \quad \text{s.t. } (2), (3), (8), (12)$$

jointly optimizes UAV trajectory, user association, and resource allocation to improve SLA stability while controlling energy consumption. The coefficients b_s, β_s, and χ are weighting parameters that control the SLA violation probability, SLA violation duration, and UAV energy consumption trade-off. Decision variables govern UAV trajectory $\mathbf{q}_u(t)$, user association $a_{k,u}(t)$, and computation resource allocation $f_{k,u}(t)$.

The problem is non-convex, stochastic, and time-coupled due to binary decisions, nonlinear rates, dynamic task arrivals, user mobility, and evolving UAV energy states. These challenges limit real-time optimal solutions and motivate efficient suboptimal and learning-based approaches. Also, purely reactive strategies, which rely only on current system observations, are insufficient in highly dynamic environments, motivating the need for predicting future tasks and mobility patterns for maintaining SLA satisfaction.

III. Proposed Predictive Multi-Agent Framework

To enable proactive control in UAV-enabled MEC systems, we propose a SLA-aware predictive multi-agent network slicing framework, depicted in Fig. 2. The proposed framework consists of three main components: (i) a prediction module that estimates user locations and task-generation status, (ii) a decentralized multi-agent decision-making module in which each UAV acts as an autonomous agent, and (iii) a centralized SLA-aware policy optimization mechanism that jointly penalizes instantaneous SLA violations, persistent slice-level degradation, predicted violations, and UAV energy consumption. The main idea is to incorporate the predictions into the decision-making process that enables UAVs to anticipate future communication and computation pressures and adjust their trajectories, user association, and computation resource allocation.

A. Prediction

This module estimates the near-future location $\hat{\mathbf{w}}_k(t+1)$ and task-generation probability $\hat{p}_k^{\lambda}(t+1)$ for each user k. It utilizes a deep RL algorithm with model $\mathcal{F}_\psi(\cdot)$, parameterized by ψ, and formulates the prediction process as a Markov decision process over the mobility-region action space. The prediction state, constructed from κ observation windows, is defined as

$$\Omega_k(t) = \{\mathbf{w}_k(t-\kappa), \lambda_k(t-\kappa), \ldots, \mathbf{w}_k(t), \lambda_k(t), s_k\}. \quad (15)$$

The prediction action $\alpha_k(t)$ is $(\hat{\mathbf{w}}_k(t+1), \hat{p}_k^{\lambda}(t+1))$. After the actual user location and task-generation status at time slot $t+1$ are observed, the prediction reward is computed as

$$\rho_k(t) = \omega_\alpha \mathbb{I}(\hat{\alpha}_k(t+1) = \alpha_k(t+1)) + \omega_\lambda \mathbb{I}(\hat{\lambda}_k(t+1) = \lambda_k(t+1)), \quad (16)$$

where ω_α and ω_λ weight mobility versus task-generation accuracy. For predicted active users, the data size $D_k(t)$ and CPU cycles $C_k(t)$ are sampled according to the slice-dependent distributions defined by the slice profile ξ_{s_k}.

We employ a Dueling Double Deep Q-learning (D3QL) architecture for the prediction model, as it improves the stability of Q-value estimation by combining double Q-learning and dueling network decomposition. The predictor's neural structure follows a hybrid recurrent-convolutional design: (i) the historical sequence is first processed by an Long Short-Term Memory (LSTM) layer to capture temporal dependencies in movement and generation behavior; (ii) then passed through convolutional layers to extract local transition patterns from the encoded sequence; and (iii) fully connected layers map the extracted features to Q-values over the prediction action space $\mathcal{A}_p$. The prediction action is selected according to an exploration-exploitation policy that selects the action with the highest Q-value $\alpha_k(t) = \arg\max_{\alpha \in \mathcal{A}_p} Q(\Omega_k(t), \alpha; \psi)$, or selects a random action to encourage exploration.

B. Decentralized Multi-agent Decision-Making

The predicted information is incorporated into a multi-agent Markov decision process, where each UAV acts as an agent. We adopt centralized training with decentralized execution: a centralized critic uses the global state to evaluate the joint behavior of UAV agents, while each UAV independently

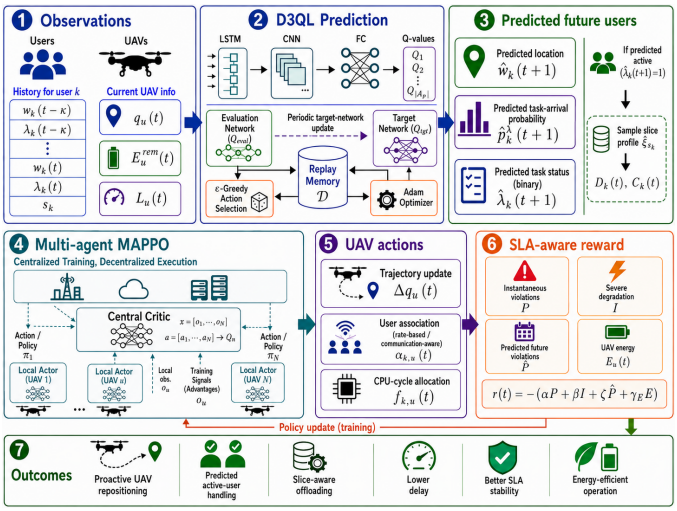

Fig. 2. Proposed framework: D3QL-based user prediction with MAPPO-based UAV trajectory control, rate-based association, and computation allocation.

selects its action using only its local observation and policy π_{θ_u}. At time slot t, the local observation of UAV u is

$$o_u(t) = \Big\{ q_u(t),\ E_u^{re}(t),\ L_u(t), \{ s_k,\ D_k(t),\ C_k(t),\ w_k(t), \hat{w}_k(t+1),\ \hat{p}_k^{\lambda}(t+1),\ \tau_k^{\max} \}_{k \in \mathcal{K}_u^{obs}(t)} \Big\}. \quad (17)$$

Here, $E_u^{re}(t)$ represents its remaining energy, $L_u(t)$ denotes its current computational load, and $\mathcal{K}_u^{obs}(t)$ shows the users that are predicted to be active and observable by u. Additionally, $\hat{w}_k(t+1)$ and $\hat{p}_k^{\lambda}(t+1)$ denote the predicted next-slot location and task-generation probability, respectively.

Each UAV agent selects an action that controls its movement and the computation resource allocation vector for the users in its observation set:

$$A_u(t) = \{\Delta q_u(t), \mathbf{f}_u(t)\},\ \mathbf{f}_u(t) = \{f_{k,u}(t)\}_{k \in \mathcal{K}_u^{obs}(t)}, \quad (18)$$

where $\Delta q_u(t)$ denotes the displacement of UAV u at time t, such that $q_u(t+1) = q_u(t) + \Delta q_u(t)$ while satisfying Eq. (2). The vector $\mathbf{f}_u(t)$ denotes the CPU-allocation vector of UAV u over its observed user set $\mathcal{K}_u^{obs}(t)$, and each element $f_{k,u}(t)$ specifies the computation resource assigned to user k. The allocated resources are constrained by the maximum computation capacity of each UAV, as given in Eq. (8). Given trajectories and channels at t, user association follows $a_{k,u}(t) = 1$ if $u = \underset{j \in \mathcal{U}}{\operatorname{argmax}} R_{k,j}(t)$ (and 0 otherwise), i.e., each user attaches to the UAV with the highest achievable rate.

C. Centralized SLA-aware Policy Optimization

We adopt MAPPO [14] to optimize the policies of UAV agents, as it supports cooperative multi-agent learning under centralized training and decentralized execution. After taking actions independently, the agents cooperate through the shared reward signal $r(t)$ designed to encourage SLA-aware and energy-efficient behavior, defined as

$$r(t) = -\Big(\textstyle\sum_{s \in \mathcal{S}} b_s P_s^{\text{viol}}(t) + \sum_{s \in \mathcal{S}} \beta_s \mathbb{I}\left(P_s^{\text{viol}}(t) > \epsilon_s\right) + \sum_{s \in \mathcal{S}} b_s \hat{P}_s^{\text{viol}}(t+1) + \chi \sum_{u \in \mathcal{U}} E_u(t) \Big). \quad (19)$$

In shared reward, $\hat{P}_s^{\text{viol}}(t{+}1)$ is the slice violation ratio computed from predicted positions $\hat{\mathbf{w}}_k(t{+}1)$, predicted activity $\hat{\lambda}_k(t{+}1)$, max-rate association under predicted geometry at $t{+}1$, and sampled $(\hat{D}_k, \hat{C}_k)$ when active; use 0 if no user is predicted active in slice s. The reward penalizes (i) instantaneous slice-level SLA violations, (ii) severe degradation to avoid persistent violating states, (iii) predicted near-future violations estimated from $\hat{\mathbf{w}}_k(t+1)$, and (iv) UAV energy consumption $E_u(t)$ with b_s, β_s, and χ controlling the trade-off between SLA satisfaction and energy efficiency.

The objective of policy optimization is to maximize the expected discounted cumulative reward $\mathbb{E}\left[\sum_{t=0}^{\infty} \mu_{\text{RL}}^t r(t)\right]$, where $\mu_{\text{RL}} \in (0,1)$ is the RL discount factor. Shared rewards enter the policy update through the temporal-difference error

$$\delta(t) = r(t) + \mu_{\text{RL}} V_\phi(\mathbf{s}(t+1)) - V_\phi(\mathbf{s}(t)), \tag{20}$$

where $V_\phi(\cdot)$ is the centralized critic and $\mathbf{s}(t) = \{o_u(t)\}_{u \in \boldsymbol{\mathcal{U}}}$ is the joint state. The advantage function is then computed via Generalized Advantage Estimation (GAE), given by

$$\hat{A}(t) = \textstyle\sum_{l=0}^{\infty} (\mu_{\text{RL}} \lambda_{\text{GAE}})^l \delta(t+l), \tag{21}$$

where λ_{GAE} is the GAE smoothing parameter. Each UAV policy is then updated using the PPO clipped surrogate objective

$$L^{\text{O}}(\theta_u) = \mathbb{E}(t)[\min(\rho_{\theta_u}(t)\hat{A}(t), \text{clp}(\rho_{\theta_u}(t), 1-\epsilon, 1+\epsilon)\hat{A}(t))], \tag{22}$$

where ϵ is the PPO clipping coefficient used to prevent excessively large policy updates and stabilize learning. Finally, the probability ratio for UAV agent u is defined by

$$\rho_{\theta_u}(t) = \frac{\pi_{\theta_u}(A_u(t)|o_u(t))}{\pi_{\theta_u^{\text{old}}}(A_u(t)|o_u(t))}. \tag{23}$$

IV. Performance Evaluation

We evaluate the proposed *predictive multi-agent slicing* framework via event-driven simulations in terms of total UAV energy consumption (propulsion + computation), average service delay, and the SLA stability metrics defined in Section II. Unless otherwise stated, we simulate $U{=}3$ UAVs serving $K{=}24$ users in a $1000{\times}1000$ m^2 area with heterogeneous HRLLC/eMBB/mMTC task profiles and slice-specific thresholds (Table II). Users follow the YJMob100K mobility traces [15], while task sizes and CPU cycles are sampled around the profiles, consistent with Section II. The proposed method leverages the predictor in Section III to incorporate $\hat{\mathbf{w}}_k(t{+}1)$ and predicted activity into the MAPPO; in all experiments, we report averages over the evaluation episodes. To perform the analysis, we conduct two scenarios.

In the first scenario, we evaluate the effectiveness of the *MAPPO* module by providing the same predicted mobility and task-generation information to all non-oracle methods: (i) *GA-Search*, a genetic search over discretized trajectory candidates; (ii) *Greedy*, which prioritizes users according to a computation-delay urgency score; and (iii) *Random*, which selects movement/allocation randomly. Therefore, the performance differences mainly reflect how each method exploits the predicted system state to guide UAV movement and compute allocation. We also include an *Offline-Optimal* oracle-style benchmark with full future information to assess the optimality gap. As shown in Fig. 3(a), increasing the number of users, which emulates user spikes, increases transmission/computation contention, leading to higher delays and pushing slices into violating regimes more frequently (higher $\bar{P}_s^{\text{viol}}$) and for longer periods (higher $\tilde{T}_s^{\text{viol}}$). Although all methods degrade with user density, MAPPO remains the most *SLA-stable* non-oracle method and stays closest to the oracle. This is because MAPPO learns a coordinated multi-UAV policy that jointly considers future user distribution, UAV energy states, and computation load, enabling proactive repositioning and SLA-aware CPU allocation. In contrast, Greedy is myopic, GA-Search is constrained by its discretized search space and finite search budget, and Random lacks SLA-aware control. Overall, MAPPO achieves lower delay and violation probability while remaining *competitive* in energy consumption, as its learned policy avoids oscillatory movements and inefficient over-provisioning; the reduced frequency of violating regimes is consistent with shorter violation persistence.

TABLE II
Simulation Parameters

Parameter	Value
Area / UAVs / users	1000×1000 m^2 / 3 / 24
UAV altitude/speed/CPU/energy	80-140m / 30mps /12GHz / $5{\times}10^7$J
Bandwidth / noise / Path-loss	8 MHz / 10^{-13} W / 2.1
HRLLC $(D_k, C_k, \tau_s^{\max}, \epsilon_s)$	(0.25 MB, 0.25 GCy, 0.08 s, 0.10)
eMBB $(D_k, C_k, \tau_s^{\max}, \epsilon_s)$	(2.2 MB, 0.75 GCy, 0.25 s, 0.20)
mMTC $(D_k, C_k, \tau_s^{\max}, \epsilon_s)$	(0.10 MB, 0.15 GCy, 0.45 s, 0.25)
Predict (LSTM/kernel,strde,pool)	128 units / (3, 2, 2)
Learning rate / μ_{RL} / λ_{GAE}	10^{-4} / 0.99 / 0.95
PPO clip / entropy coeff.	0.2 / 0.02
Train & eval episode/Hidd. dim	1500, 10 / 128
GA Populate/Generate/Mutation	24 / 25 / 0.08

In the second scenario, we evaluate the prediction module by comparing *Predictive-MAPPO* with *Informed-MAPPO*, where the latter uses full future mobility information and serves as an upper-bound reference for prediction quality. Fig. 3(b) shows that increasing the number of UAVs, which emulates resource sufficiency, significantly reduces delay and SLA violation probability for both methods by improving spatial coverage, shortening user-UAV distances, and increasing computation capacity. The gap between Predictive-MAPPO and Informed-MAPPO is more visible with fewer UAVs, as prediction errors are more harmful under scarce aerial resources and cause suboptimal repositioning or computation bottlenecks. However, as the number of UAVs increases, Predictive-MAPPO closely approaches Informed-MAPPO in delay and SLA violation probability, showing that the proposed predictor provides *sufficiently accurate* future information for proactive slicing. Energy consumption increases for both methods as more UAVs participate in movement and computation, while their energy curves remain *almost aligned*. This indicates that the gain of Informed-MAPPO mainly comes from more accurate anticipation rather than excessive energy use, confirming that Predictive-MAPPO achieves near-informed SLA-aware control using only learned predictions.

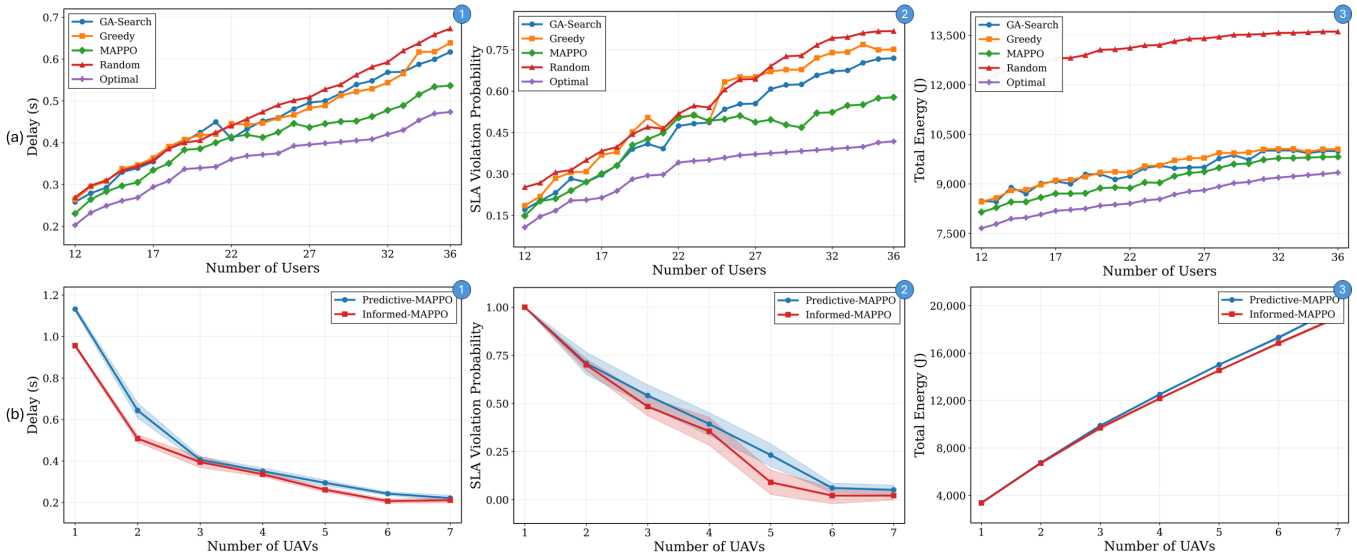

Fig. 3. Performance under (a) increasing number of users and (b) increasing number of UAVs: (1) Delay, (2) SLA violation probability, and (3) UAV energy.

V. Conclusion

In this paper, we studied SLA-aware network slicing for UAV-enabled MEC under user mobility, stochastic task arrivals, and limited onboard energy as well as computing resources. To address the resulting stochastic, non-convex, and time-coupled control problem, we proposed a predictive multi-agent framework in which cooperative UAV agents are trained with MAPPO under centralized training and decentralized execution, utilizing mobility and task-generation predictions to act proactively. Simulation results showed that the proposed Predictive-MAPPO improves SLA stability (lower violation probability and shorter violation duration) while remaining competitive in energy consumption and delay performance compared with baselines, and approaches the oracle benchmark with sufficiently accurate predicted information. Future work will incorporate more realistic propulsion and interference models; treat uplink transmit power explicitly as an optimization variable for joint power, trajectory, association, and computation control; and consider dynamic slice admission control as well as adaptive bandwidth allocation. Moreover, we will explore LLM-driven *agentic* orchestration [16] and semantic-aware control [17] for UAV-enabled slicing, combining high-level planning with continual learning and semantics-oriented reward feedback.

Acknowledgment

The research work is supported in part by the Federal Ministry of Research, Technology, and Space (BMFTR), Germany, through the Project 6GEM+ under Grant 16KIS2411; the European Union's Horizon Europe research and innovation programme under the 6G-Path project (Grant No. 101139172); and the Research Council of Finland 6G Flagship Programme under Grant No. 369116.

References

[1] M. Farhoudi, M. Shokrnezhad, T. Taleb, R. Li, and J. Song, "Discovery of 6G services and resources in edge-cloud-continuum," *IEEE Netw.*, vol. 39, no. 3, pp. 223–232, 2025.

[2] S. Barick and C. Singhal, "UAV-assisted MEC architecture for collaborative task offloading in urban IoT environment," *IEEE Trans. Netw. Service Manag.*, vol. 22, no. 1, pp. 732–743, 2025.

[3] Z. Sasan and S. Khorsandi, "Balancing resource utilization and slice dissatisfaction through dynamic soft slicing for 6G wireless networks," *Scientific Reports*, vol. 15, no. 1, p. 22987, 2025.

[4] Z. Sasan *et al.*, "Joint network slicing, routing, and in-network computing for energy-efficient 6G," in *Proc. IEEE Wireless Commun. and Networking Conf.* IEEE, 2024, pp. 1–6.

[5] M. Farhoudi *et al.*, "Deep learning based service composition in integrated aerial-terrestrial networks," in *Proc. IEEE Int. Conf. Netw. Softwarization*. IEEE, 2025, pp. 204–208.

[6] M. Farhoudi, H. Mazandarani, M. Shokrnezhad, T. Taleb, and I. Lacalle, "Energy efficient orchestration in multiple-access vehicular aerial-terrestrial 6G networks," *IEEE Trans. Veh. Technol.*, 2026.

[7] G. Wu, B. Zhang, and Y. Li, "Intelligent and survivable resource slicing for 6G-oriented UAV-assisted edge computing networks," *Computer Commun.*, vol. 202, pp. 154–165, 2023.

[8] J. Tang, J. Nie, J. Zhao, Y. Zhou, Z. Xiong, and M. Guizani, "Slicing-based software-defined mobile edge computing in the air," *IEEE Wireless Commun.*, vol. 29, no. 1, pp. 119–125, 2022.

[9] G. Faraci, C. Grasso, and G. Schembra, "Design of a 5G network slice extension with MEC UAVs managed with reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2356–2371, 2020.

[10] J. Tian, D. Wang, H. Zhang, and D. Wu, "Service satisfaction-oriented task offloading and UAV scheduling in UAV-enabled MEC networks," *IEEE Trans. Wireless Commun.*, vol. 22, no. 12, pp. 8949–8964, 2023.

[11] P. Chen *et al.*, "QoS-oriented task offloading in NOMA-based multi-UAV cooperative MEC systems," *IEEE Trans. Wireless Commun.*, 2025.

[12] X. Li *et al.*, "Self-adjusting network slicing for dynamic heterogeneous task offloading in UAV-enabled mobile edge computing," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 12, pp. 673–687, 2026.

[13] ITU-R, "Framework and overall objectives of the future development of IMT for 2030 and beyond," International Telecommunication Union, Radiocommunication Sector, Tech. Rep. M.2160-0, Nov. 2023.

[14] C. Yu *et al.*, "The surprising effectiveness of PPO in cooperative, multi-agent games," 2022. [Online]. Available: https://arxiv.org/abs/2103.01955

[15] T. Yabe *et al.*, "YJMob100K: City-scale and longitudinal dataset of anonymized human mobility trajectories," *Scientific Data*, vol. 11, no. 1, p. 397, 2024.

[16] M. Shokrnezhad and T. Taleb, "An autonomous network orchestration framework integrating large language models with continual reinforcement learning," *IEEE Commun. Mag.*, vol. 63, no. 8, pp. 78–84, 2025.

[17] H. Mazandarani *et al.*, "Semantic-aware dynamic and distributed power allocation: a multi-UAV area coverage use case," in *Proc. Int. Conf. Mach. Learn. for Commun. and Netw.*, 2025, pp. 1–6.