# Place, Slice and Schedule: Hierarchical O-RAN Control of a Tethered mmWave UAV-gNB

Alireza Mohammadhosseini and Fatemeh Afghah
*Department of Electrical and Computer Engineering, Clemson University, Clemson, SC, USA*

***Abstract*—Unmanned aerial vehicle (UAV)-mounted 5G New Radio base stations (gNBs) can augment terrestrial networks with an on-demand, repositionable Frequency Range 2 (FR2) capacity layer. This flexibility, however, couples the physical network topology with radio-resource management: UAV movement reshapes blockage, channel quality, and the set of effectively served users, while traffic demand, queues, and service requirements evolve at a much faster timescale. Existing Open Radio Access Network (O-RAN)-enabled UAV studies optimize trajectory, deployment, association, or resource allocation, but typically in isolation, without coordinating slow aerial control with fast per-user scheduling. We instead exploit O-RAN disaggregation, Key Performance Indicator (KPI) monitoring, and multi-timescale RAN Intelligent Controller (RIC) control to address this coupling: a Non-Real-Time RIC rApp uses aggregated KPIs and radio-environment context to jointly control tethered UAV placement and the enhanced Mobile Broadband (eMBB)/Ultra-Reliable Low-Latency Communication (URLLC) slice budget, while a Near-Real-Time RIC xApp allocates per-user resources within that budget. We realize this xApp as a permutation-equivariant *DeepSets Soft Actor-Critic (D-SAC) scheduler* that treats the users as an unordered set, trained in a Sionna RT ray-traced channel. The resulting hierarchical controller improves eMBB SLA satisfaction by up to 17% and URLLC on-time delivery by up to 42% over classical and learned schedulers; the learned rApp further raises URLLC on-time delivery by up to 20% over baselines.**

## I. Introduction

UAV-mounted base stations have emerged as a flexible way to add capacity and coverage on demand [1], and tethered UAVs have been introduced to provide persistent power and remove the flight-time limit [2]. Serving such aerial cells well requires programmable, adaptive control of both where the UAV is placed and how it allocates radio resources to meet the service requirements of 5G-advanced and 6G networks. Open Radio Access Network (O-RAN) provides this through a disaggregated, programmable architecture whose control logic is exposed by the RAN Intelligent Controller (RIC) [3]: control is organized by timescale, with a non-real time RIC (Non-RT RIC) hosting rApps that generate high-level policy over the A1 interface at second-level timescales, and a near-real time RIC (Near-RT RIC) hosting xApps that act over E2 within 10 ms–1 s. O-RAN is emerging as a programmable control architecture for non-terrestrial RAN, enabling RIC-based optimization of mobility, radio resources, and service continuity across multiple timescales [4].

These decisions span timescales: slow placement and slice-budget choices shape coverage and long-term service satisfaction, while fast per-user allocation decides whether individual service-level agreements (SLAs) are met. Because a decision at one layer propagates to the other and changes quality-of-service (QoS) or reliability outcomes [5], the two loops cannot be tuned independently, especially at millimeter-wave frequencies, where the UAV position determines per-user coverage through line-of-sight blockage.

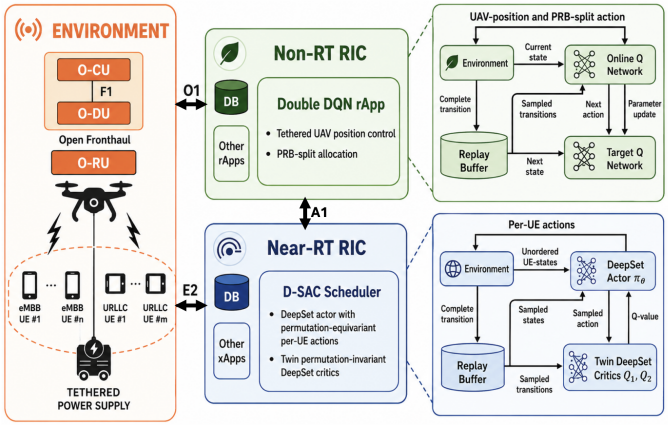



Fig. 1: Hierarchical O-RAN control of the tethered mmWave UAV-gNB: the rApp sets UAV 3D-placement and the slice PRB budget, and the D-SAC xApp allocates PRBs per user.

A second challenge arises from the structure of near-real-time resource control. As the number of active UEs grows, a flat multilayer perceptron (MLP) policy is tied to a fixed input dimension and UE ordering, although UE indices have no physical meaning and the active-user population may vary over time; such policies neither preserve scheduling decisions under UE permutations nor transfer to different user counts without retraining. Graph-based RL [6] can handle variable-size observations, but requires constructing an explicit user graph and message passing, even when users interact only through competition for a common resource budget. We therefore model the active UE population as an unordered set and design a permutation-equivariant, parameter-shared policy, which provides structural scalability while retaining a fixed architecture as the number and ordering of UEs change.

We propose a hierarchical O-RAN controller for a tethered UAV-gNB that co-designs both loops, illustrated in Fig. 1. The Non-RT rApp selects the tethered UAV placement together with the eMBB/URLLC slice PRB split, handed to the Near-RT xApp as an A1 policy; conditioned on that policy, the xApp performs per-user PRB allocation every radio slot. We realize the xApp as a permutation-equivariant *DeepSets SAC scheduler* (D-SAC) [7], [8] and decouple the loops with a

This material is based upon work supported by the National Science Foundation under Grant Numbers CNS-2202972, CNS-2318726, and CNS-2232048

staged procedure: the scheduler is trained under randomized high-level policy, the rApp is trained over the frozen scheduler, and the scheduler is fine-tuned to the rApp policy. Channel gains for every tethered UAV candidate are precomputed with Sionna RT ray tracing [9], capturing the line-of-sight and blockage geometry that drives mmWave coverage rather than relying on a stochastic channel model. Our contributions are:

- A hierarchical two-timescale intelligent O-RAN controller for a tethered mmWave UAV-gNB in which a rApp sets UAV 3D-placement and slice PRB budget and a xApp performs conditional per-user scheduling, together with a staged co-design procedure that decouples the two RIC loops.
- A scalable permutation-equivariant DeepSets Soft Actor-Critic (D-SAC) scheduler for variable-cardinality UE populations. Shared per-UE processing and permutation-invariant global pooling remove dependence on UE indexing, enable a single policy architecture to operate across different user counts, and avoid the explicit graph construction required by graph-based schedulers.
- A geometry-aware evaluation using Sionna RT ray-traced FR2 channels demonstrating that the hierarchical controller improves both eMBB SLA satisfaction and URLLC on-time delivery over classical and learned baselines, while the same D-SAC policy retains its performance advantage as the UE population increases without retraining.

## II. Related Work

UAV placement has been studied for guaranteed mmWave line-of-sight coverage [10], for integrated access and backhaul [11], and for joint positioning and association via deep Q-learning [12], while tethered UAVs have been analyzed for coverage [2] and controlled via multi-agent Q-learning [13]. However, UAV placement is commonly optimized with respect to coverage, association, throughput, or another aggregate network objective while the fast per-user scheduler is fixed or abstracted. This misses an important coupling: each UAV movement changes the channel state and feasible service region available to the scheduler, while heterogeneous user traffic and service-level requirements determine which UAV positions are desirable in the first place.

The O-RAN architecture provides a programmable framework for exposing these coupled control variables through RIC loops operating at different timescales. U-ORAN [14] introduced RIC-based control for multi-UAV networks and jointly optimized UAV trajectories and task offloading, while [15] jointly considered aerial radio-unit deployment, user association, and resource allocation in an O-RAN-enabled UAV network. These works demonstrate the value of O-RAN for intelligent aerial-network control, but the radio-resource decisions remain primarily coupled to trajectory, association, or offloading objectives rather than to a service-aware per-user scheduler operating at a distinct faster control timescale. Separately, terrestrial O-RAN research has investigated learning-based radio control: xSlice [6] employs actor–critic deep RL with a graph representation to support a varying number of traffic sessions, CollabORAN [16] coordinates resource control across multiple RIC timescales, and [17] considers learning-based per-UE RAN parameter adaptation, while full-stack xApp implementations on srsRAN demonstrate closed-loop RL slicing [18] and meta-RL improves adaptation to nonstationary O-RAN traffic [19]. These studies establish the feasibility of multi-timescale and intelligent control in O-RAN, but do not address the additional feedback loop introduced when the gNB itself is a controllable aerial platform.

When UEs mainly compete for a shared resource budget, a set-based representation provides a simpler alternative to graph-based scheduling: we represent the active UE population as an unordered set and use a permutation-equivariant DeepSets policy whose encoder and decision layers are shared across users and whose global context is obtained through permutation-invariant pooling, yielding a single parameterization that is independent of UE ordering and operates across different UE counts. Different from the above lines of work, we jointly address the two structural challenges introduced by an aerial O-RAN: *multi-timescale coupling between physical topology and radio-resource control*, and *variable-cardinality per-user scheduling*.

## III. System Model

We consider a tethered UAV-gNB serving mobile eMBB and URLLC users over a ray-traced mmWave downlink, controlled by two coupled O-RAN layers: a slow Non-RT rApp that selects the UAV 3D-placement and the slice-level PRB budget, and a fast Near-RT xApp that allocates the slice PRB budgets among individual UEs conditioned on that policy. In our O-RAN system, the rApp generates the A1 policy, and the xApp enforces its per-UE PRB decisions through E2SM-RC while consuming per-slot KPIs via E2SM-KPM over E2. The rApp state is built from O1 performance-management telemetry, supplied as R1 enrichment information.

### A. Channel Model

The UAV-gNB is tethered to a fixed ground anchor and can only occupy positions in a feasible tethered candidate set $\mathcal{V}$. A generic tethered position can be parameterized by the cable length $L_{\mathrm{te}}$, an elevation angle $\rho$ measured from the vertical axis, and an azimuth angle $\phi$:

$$\mathbf{v}(\rho,\phi) = \mathbf{v}_0 + L_{\mathrm{te}} \begin{bmatrix} \sin\rho\cos\phi \\ \sin\rho\sin\phi \\ \cos\rho \end{bmatrix}, \tag{1}$$

where $\mathbf{v}_0$ is the ground anchor. In the control problem, the rApp selects a neighboring candidate $\mathbf{v}_{t+1} \in \mathcal{N}_{\mathrm{te}}(\mathbf{v}_t)$ from the tether-feasible navigation graph.

Let $\mathbf{x}_{u,t}$ be the position of UE $u$ at step $t$, $\mathbf{v}_t \in \mathcal{V}$ the selected UAV candidate, and $\beta(\mathbf{v}_t, \mathbf{x}_{u,t})$ the location-dependent power gain obtained from the Sionna RT radio map. The system has $N$ PRBs of bandwidth $B_{\mathrm{PRB}}$. We model a single UAV-gNB downlink cell, so link quality is represented by the downlink SNR of each UE. With uniform power allocation across PRBs, the transmit power per PRB is $P_{\mathrm{tx}}/N$, and the downlink SNR of UE $u$ in slot $\ell$ of rApp step $t$ is:

$$\gamma_{u,t,\ell} = \frac{(P_{\mathrm{tx}}/N)\beta(\mathbf{v}_t, \mathbf{x}_{u,t})h_{u,t,\ell}}{N_0 B_{\mathrm{PRB}}}, \tag{2}$$

where $P_{\mathrm{tx}}$ is total transmit power, $h_{u,t,\ell}$ is an independent unit-mean small-scale fading power gain capturing slot-scale variations not represented by the precomputed radio map, and

$N_0$ is the noise spectral density including receiver noise figure. If UE $u$ receives $n_{u,t,\ell}$ PRBs, its downlink rate is $R_{u,t,\ell} = n_{u,t,\ell} B_{\mathrm{PRB}} \log_2(1+\gamma_{u,t,\ell})$.

### B. Service Model and QoS Metrics

The UAV-gNB serves eMBB UEs $\mathcal{U}_e$ and URLLC UEs $\mathcal{U}_u$, with $\mathcal{U}_e \cup \mathcal{U}_u = \mathcal{U}$. The rApp selected budget $N_t^e$ limits the total eMBB PRBs available in control slot $t$, while the xApp decides the per-UE allocations $n_{u,t,\ell}$ at each radio slot. The mean eMBB throughput over the $L$ radio intervals of control window $t$ is $\bar{R}_t^e = \frac{1}{L|\mathcal{U}_e|}\sum_{\ell=0}^{L-1}\sum_{u\in\mathcal{U}_e} R_{u,t,\ell}$, which we map to a bounded eMBB QoS score:

$$q_t^e = \min\left(\frac{\bar{R}_t^e}{R_{\mathrm{target}}}, 1\right), \tag{3}$$

where $R_{\mathrm{target}}$ is a network-level reference rate that normalizes this aggregate reward-shaping term. The eMBB SLA is enforced per UE, with a violation whenever a UE's throughput falls below its own minimum-rate guarantee $R_{u,\min}$. The eMBB violation ratio is:

$$\mathcal{V}_t^e = \frac{1}{L|\mathcal{U}_e|}\sum_{\ell=0}^{L-1}\sum_{u\in\mathcal{U}_e} \mathbb{I}[R_{u,t,\ell} < R_{u,\min}]\,. \tag{4}$$

URLLC is modeled as packet-based, deadline-constrained GBR traffic with per-UE packet delay budgets in the tens of milliseconds, and on-time delivery is reported as a scheduling metric under an intentionally overloaded regime rather than as a reliability figure. For URLLC UE $u$, packet arrivals in radio interval $\ell$ follow $A_{u,t,\ell} \sim \mathrm{Poisson}(\lambda_u \Delta)$, where $\lambda_u$ is the arrival rate and $\Delta$ is the radio slot duration. Packets are queued per UE and may be retransmitted while their deadlines and retransmission budgets permit, so the queued bits evolve as:

$$Q_{u,t,\ell+1} = \max(0, Q_{u,t,\ell} + b_u A_{u,t,\ell} - S_{u,t,\ell}), \tag{5}$$

where $b_u$ is packet size in bits and $S_{u,t,\ell}$ is the service in bits, bounded by capacity $C_{u,t,\ell} = R_{u,t,\ell}\,\Delta$ and by the packet decoding outcome. A packet misses its deadline if it is not delivered within $D_u$ seconds of arrival, giving the window-level deadline-miss rate $\mathcal{V}_t^u = N_t^{\mathrm{miss}} / \max(1, N_t^{\mathrm{del}} + N_t^{\mathrm{miss}})$, where $N_t^{\mathrm{del}}$ and $N_t^{\mathrm{miss}}$ are the number of delivered and missed packets in window $t$. With the mean URLLC queue $\bar{Q}_t^u = \frac{1}{L|\mathcal{U}_u|}\sum_\ell \sum_{u\in\mathcal{U}_u} Q_{u,t,\ell}$, we define URLLC QoS as:

$$q_t^u = 1 - \frac{\bar{Q}_t^u}{\bar{Q}_t^u + Q_{\mathrm{ref}}}, \tag{6}$$

where $Q_{\mathrm{ref}}$ is the queue level at which the queue-pressure term reaches one half. Thus $q_t^u$ provides QoS feedback before deadlines expire, while $\mathcal{V}_t^u$ measures SLA violation.

## IV. Problem Formulation

We formulate the controller as a hierarchical MDP, splitting control into a slow rApp decision and a fast xApp scheduling decision. The rApp acts only on O-RAN aggregate KPIs and does not require per-slot per-UE MAC (Media Access Control) state, whereas the xApp acts on the finer per-UE radio-control information.

### A. Hierarchical MDP

At rApp step $t$, the slow-timescale MDP is $\mathcal{M}_R = (\mathcal{S}_R, \mathcal{A}_R, P_R, r_R, \Gamma_R)$. The rApp observes $s_t^R \in \mathcal{S}_R$ and chooses a joint action $a_t^R = (\alpha_t, \eta_t) \in \mathcal{A}_R$, where $\alpha_t$ is a tethered UAV movement command and $\eta_t = (\eta_t^e, \eta_t^u) \in \mathcal{H}$ is the eMBB/URLLC slice PRB split from a finite split set $\mathcal{H}$. The movement command applies a single-axis step $\alpha_t \in \{\texttt{stay}, \rho^+, \rho^-, \phi^+, \phi^-\}$; these primitives generate exactly the tether-feasible neighborhood $\mathcal{N}_{\mathrm{te}}(\mathbf{v}_t)$ of (1), restricting movement to one grid hop per control step. The chosen split fixes the per-slice PRB budgets $N_t^e = \eta_t^e N$ and $N_t^u = \eta_t^u N$ for the following rApp window. The state $s_t^R$ is a fixed-length vector of window-aggregated features: the normalized UAV placement and previous movement, the previous slice split, per-slice UE spatial and radio-quality summaries, QoS-pressure and slice-composition statistics, and a candidate-lookahead block summarizing the per-slice service each neighboring position would provide. All entries are aggregate measurements over the previous 1 s window, consistent with the KPI-level information a Non-RT RIC observes rather than per-slot MAC scheduler state.

Within the rApp window, the xApp problem is modeled as the MDP $\mathcal{M}_X(a_t^R) = (\mathcal{S}_X, \mathcal{A}_X, P_X, r_X, \Gamma_X \mid a_t^R)$. At radio slot $\ell$, it observes a per-UE feature matrix $\mathbf{F}_{t,\ell} = [f_{1,t,\ell}, \ldots, f_{|\mathcal{U}|,t,\ell}]^\mathsf{T}$, where each row $f_{u,t,\ell} \in \mathbb{R}^{12}$ combines the channel, queue, throughput, and deadline/SLA state of UE $u$ with the rApp slice-budget context. The xApp emits one continuous action per UE, $\mathbf{a}_{t,\ell}^X = [a_{1,t,\ell}^X, \ldots, a_{|\mathcal{U}|,t,\ell}^X]^\mathsf{T}$, which is mapped to a feasible PRB allocation $\mathbf{n}_{t,\ell}$ inside the rApp selected slice budgets. For each slice $s \in \{e, u\}$:

$$\sum_{u\in\mathcal{U}_s} n_{u,t,\ell} \le N_t^s, \qquad n_{u,t,\ell} \ge 0. \tag{7}$$

This continuous PRB-share relaxation represents average PRB shares over the radio-control interval. Integer PRB rounding is outside the present model. The resulting hierarchical policy factorizes as:

$$\pi(a_t^R, \mathbf{a}_{t,0:L-1}^X) = \pi_R(a_t^R | s_t^R) \prod_{\ell=0}^{L-1} \pi_X(\mathbf{a}_{t,\ell}^X | \mathbf{F}_{t,\ell}, a_t^R), \tag{8}$$

which makes the dependency explicit: the xApp policy is conditioned on the rApp intent through the slice-budget context and the resulting channel/queue state. In both tuples, $P$ is the induced transition kernel and $\Gamma \in (0,1)$ the discount factor.

### B. Objective and Reward

The window-level QoS score is the weighted sum of eMBB QoS in (3) and URLLC QoS in (6), $U_t = w_e^{\mathrm{qos}} q_t^e + w_u^{\mathrm{qos}} q_t^u$. There is no movement-energy term because the UAV is tethered and powered through the cable; mobility is constrained through the feasible tether graph instead of penalized by an

energy model. For an episode horizon of $T$ rApp steps, the joint hierarchical objective is:

$$\max_{\pi_R,\pi_X} \quad \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}_{\pi_R,\pi_X}[U_t] \tag{9a}$$

$$\text{s.t.} \quad \frac{1}{T}\sum_{t}\mathbb{E}[\mathcal{V}_t^e]\le\epsilon_e, \quad \frac{1}{T}\sum_{t}\mathbb{E}[\mathcal{V}_t^u]\le\epsilon_u, \tag{9b}$$

$$\mathbf{v}_{t+1}\in\mathcal{N}_{\text{te}}(\mathbf{v}_t), \quad \eta_t^e+\eta_t^u=1, \tag{9c}$$

$$\eta_t^e,\ \eta_t^u,\ n_{u,t,\ell}\ge 0, \quad \forall u,t,\ell, \tag{9d}$$

where $\mathcal{N}_{\text{te}}(\mathbf{v}_t)$ is the tether-feasible neighbor set and $\epsilon_e,\epsilon_u$ denote SLA violation limits. For training, we relax the constrained problem into an unconstrained per-step reward with fixed penalty weights $\kappa_e,\kappa_u>0$, i.e. a Lagrangian relaxation with fixed multipliers:

$$r_t = U_t-\kappa_e\mathcal{V}_t^e-\kappa_u\mathcal{V}_t^u. \tag{10}$$

## V. Proposed Solution

End-to-end optimization of (9) is computationally expensive because one rApp action induces many xApp decisions, so we use a staged training procedure that preserves the hierarchy in (8). First, the xApp is trained as a conditional fast-timescale scheduler under randomized UAV candidates and randomized slice budgets. Second, the xApp is frozen and the rApp is trained on the induced slow-timescale MDP. Third, the xApp is fine-tuned with the learned rApp frozen, which adapts the near-RT scheduler to the state distribution induced by the rApp; all reported D-SAC results use this fine-tuned scheduler.

### A. Per-UE PRB Scheduling

For fixed rApp state and action, the UAV candidate and slice budgets are fixed during the rApp control window, and the xApp controls only the per-UE allocation vector $\mathbf{n}_{t,\ell}=[n_{1,t,\ell},\dots,n_{|\mathcal{U}|,t,\ell}]^\mathsf{T}$ for each slot $\ell$, subject to the per-slice budgets of (7). The conditional xApp training problem is:

$$\max_{\pi_X}\ \mathbb{E}_{\pi_X}\left[\sum_{\ell=0}^{L-1} r_X(\mathbf{F}_{t,\ell},\mathbf{n}_{t,\ell})\mid s_t^R,a_t^R\right], \tag{11}$$

where the learning reward $r_X$ is the fast timescale counterpart of (10): it rewards eMBB QoS and URLLC queue QoS, while penalizing eMBB SLA violation and URLLC deadline miss to provide dense scheduling feedback.

### B. DeepSets SAC Scheduler (D-SAC)

The xApp is trained with Soft Actor-Critic (SAC) [8]. A flat MLP actor would tie parameters to UE indices, which is undesirable for scheduling because if the UE order is permuted, the output scores should be permuted in the same way. Therefore, we keep SAC as the learning algorithm but parameterize its actor and critic with *DeepSets* networks [7]. Unlike a GNN-based scheduler [20], this design does not require constructing an explicit graph or performing message passing, which suits our setting because the xApp must map an unordered UE set to per-UE PRB priority scores, requiring permutation equivariance rather than explicit relational inference.

The actor applies shared per-UE layers $\mathbf{h}_{u,t,\ell}=\phi_\theta(f_{u,t,\ell})$ to each feature vector, then forms a permutation-invariant global context using mean and max pooling:

$$\mathbf{c}_{t,\ell}=\psi_\theta\Big(\frac{1}{|\mathcal{U}|}\sum_{j\in\mathcal{U}}\mathbf{h}_{j,t,\ell},\ \max_{j\in\mathcal{U}}\mathbf{h}_{j,t,\ell}\Big). \tag{12}$$

Mean pooling captures the average traffic and channel condition, while max pooling captures extreme cases such as highly urgent or backlogged UEs. A shared stochastic SAC head $g_\theta$ then maps each UE's embedding and the global context to a per-UE action $a_{u,t,\ell}=\tanh(\mu_{u,t,\ell}+\sigma_{u,t,\ell}\,\epsilon_{u,t,\ell})\in(-1,1)$, with $(\mu_{u,t,\ell},\sigma_{u,t,\ell})=g_\theta(\mathbf{h}_{u,t,\ell},\mathbf{c}_{t,\ell})$ and SAC reparameterization noise $\epsilon_{u,t,\ell}\sim\mathcal{N}(0,1)$. Because all per-UE layers are shared and the context is permutation invariant, the actor is *permutation equivariant* for any UE permutation matrix. The bounded actions are mapped to strictly positive scheduling priorities by a centered softplus:

$$z_{u,t,\ell}=\text{softplus}\left(a_{u,t,\ell}-\frac{1}{|\mathcal{U}|}\sum_{j\in\mathcal{U}}a_{j,t,\ell}\right), \tag{13}$$

which keeps the priorities positive and differentiable while centering them so that the per-slice normalization below responds to relative rather than absolute action values. The priorities are normalized within each slice to produce feasible PRB shares:

$$n_{u,t,\ell}=\begin{cases}\eta_t^e N\dfrac{z_{u,t,\ell}}{\sum_{j\in\mathcal{U}_e}z_{j,t,\ell}}, & u\in\mathcal{U}_e,\\[2ex] \eta_t^u N\dfrac{z_{u,t,\ell}}{\sum_{j\in\mathcal{U}_u}z_{j,t,\ell}}, & u\in\mathcal{U}_u.\end{cases} \tag{14}$$

The critic evaluates the whole allocation with one scalar Q-value, so it must be *permutation invariant*. Each SAC twin critic applies shared layers $\mathbf{y}_{u,t,\ell}=\varphi_\omega(f_{u,t,\ell},a_{u,t,\ell})$ to each per-UE feature/action pair, then pools over UEs and predicts $Q_\omega(\mathbf{F}_{t,\ell},\mathbf{a}_{t,\ell})=q_\omega\big(\frac{1}{|\mathcal{U}|}\sum_{j\in\mathcal{U}}\mathbf{y}_{j,t,\ell},\max_{j\in\mathcal{U}}\mathbf{y}_{j,t,\ell}\big)$. Thus, the actor maps an unordered UE set to per-UE scores equivariantly, while the critic maps the unordered set of UE feature/action pairs to one invariant value.

### C. rApp Training over the Frozen xApp

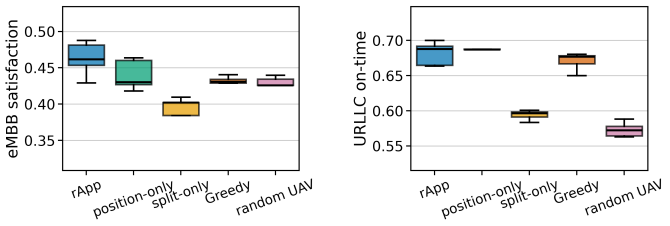
In the second stage, $\pi_X$ is frozen and absorbed into the environment: the frozen per-UE scheduler, together with UE mobility, packet arrivals, and channel variation, forms the transition kernel of the slow-timescale MDP $\mathcal{M}_R$ of Sec. IV-A. This lets the rApp be trained as a standard MDP over its aggregate O-RAN state, with Double DQN [21] over the discrete joint action set of tether-movement commands and slice-budget splits detailed in Sec. VI.

## VI. Experimental Setup

The deployment area is $200\,\text{m}\times 200\,\text{m}$ and the UAV-gNB is tethered to the map center by a fixed $100\,\text{m}$ cable. The candidate set uses five elevation levels $\{0,12.5,25,37.5,50\}^\circ$ and five azimuth levels $\{0,72,144,216,288\}^\circ$, giving 21 geometrically distinct positions over an altitude range of $64.3$–$100\,\text{m}$; the corresponding channel-gain maps are precomputed with Sionna RT at $28\,\text{GHz}$ (n257). At each step the rApp selects a slice split from

TABLE I: Simulation and training parameters.

| Parameter | Value |
|---|---|
| Carrier frequency / band | 28 GHz (n257) |
| PRBs / PRB bandwidth | 132 / 1.44 MHz |
| Tx power / rApp, xApp slots | 36 dBm / 1 s, 10 ms |
| UEs / eMBB:URLLC ratio | $30, 40, 50$ / $0.6 : 0.4$ |
| Episode length | 100 steps |
| eMBB $R_{\text{target}}$ / $R_{u,\min}$ | 10 / $[2, 70]$ Mbps |
| URLLC packet / arrival | 256 B / $[50, 800]$ pkt/s |
| URLLC deadline $D_u$ | $[20, 50]$ ms |
| Penalties $(\kappa_e, \kappa_u)$ / $\Gamma$ | $(3.0, 3.0)$ / 0.99 |
| D-SAC encoder / context / critic | 64 / 64 / 256 |
| DDQN layers / xApp,rApp LR | $[256, 256]$ / $3{\times}10^{-4}, 10^{-4}$ |
| Training steps (Phase 1/2/3) | 300k / 150k / 100k |
| Evaluation seeds / episodes | 5 / 50 |




(a) eMBB SLA satisfaction (b) URLLC on-time delivery

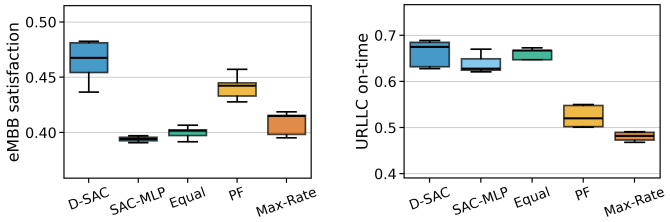
Fig. 2: Contribution of the rApp. The rApp improves eMBB SLA satisfaction and URLLC on-time delivery over baselines.

$\{(0.9, 0.1), (0.7, 0.3), (0.5, 0.5), (0.3, 0.7), (0.1, 0.9)\}$, and UEs move at $1\,\mathrm{m/s}$ under a uniform random walk. Remaining parameters are in Table I.

eMBB UEs are assigned different base loads and minimum-rate SLAs to emulate the heterogeneous experienced data rate requirements of 5G eMBB service tiers, and URLLC UEs different arrival rates and deadlines, creating unequal per-UE service pressure within each slice. These wide ranges place the system in a demanding regime where some eMBB minimum-rate targets exceed the per-UE capacity of the cell and part of the service area falls in ray-traced blockage where the channel gain drops to the noise floor. Therefore, eMBB SLA satisfaction and URLLC on-time delivery are bounded by the SLA and coverage geometry; the residual headroom is what the two control loops compete over, and we report them as comparative metrics under identical conditions. All results report the mean and 95% confidence interval over five random seeds. A per-slot allocation by the D-SAC scheduler takes 0.12 ms for 30 users on a single CPU thread (about 1% of the 10 ms radio-slot budget), confirming Near-RT feasibility.

## VII. Results

We evaluate the fully trained hierarchical controller of Sec. V. To characterize service quality we report two per-UE metrics instead of raw violation rates. The eMBB SLA satisfaction ratio $\bar{s}^e = \frac{1}{|\mathcal{U}_e|} \sum_{u \in \mathcal{U}_e} \min\left(\frac{\bar{R}_u}{R_{u,\min}}, 1\right)$, where $\bar{R}_u$ is UE $u$'s mean rate over the evaluation window, is the average fraction of each eMBB UE's contracted minimum rate that is delivered, and the URLLC on-time delivery ratio $\bar{o}^u = 1 - \mathcal{V}^u$ is the fraction of URLLC packets delivered within their deadline. We additionally report aggregate throughput and Jain's fairness [22].

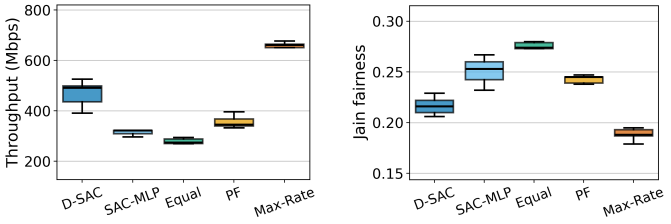



(a) eMBB SLA satisfaction (b) URLLC on-time delivery

Fig. 3: Performance comparison; D-SAC outperforms all classical and learned baselines.

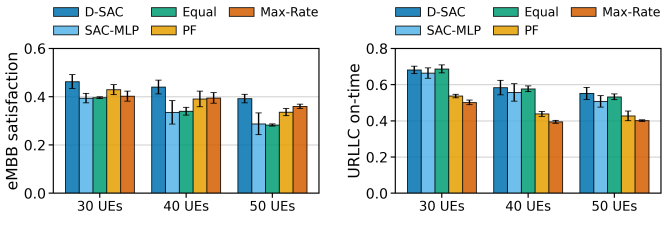
*1) Performance Evaluation of the proposed rApp:* Fig. 2 isolates the rApp by varying only the high-level policy while keeping the same low-level D-SAC scheduler. Alongside the full learned rApp we include two ablations that expose the marginal value of each rApp decision: *rApp position-only* applies only the learned UAV placement decision with fixed slice split, and *rApp split-only* applies only the learned slice split while holding the UAV at the map center. We further compare against a *greedy* method that at each step moves the UAV toward the reachable tether candidate with the best predicted cover age from the radio map, and a random-movement UAV; both use a fixed split. Two effects stand out. First, learned placement is the dominant contributor: rApp move-only alone reaches 0.440 eMBB satisfaction and 0.687 URLLC on-time, far above the pinned rApp split-only (0.391, 0.594), and the learned rApp surpasses even the strong greedy method on both axes. Second, learned slice control adds a distinct gain on top of placement: the full rApp lifts eMBB satisfaction from position-only 0.440 to 0.463 by shifting budget toward eMBB, trading only a negligible URLLC on-time (0.682 versus 0.687) for that improvement. Relative to the random baseline the learned rApp improves URLLC on-time delivery by up to 20% and eMBB satisfaction by up to 12%. Since the low-level D-SAC scheduler is identical across all arms, these differences reflect the learned high-level policy.

*2) Scheduler Comparison:* In Fig. 3 we compare the proposed D-SAC scheduler with the classical proportional-fair (PF), Equal, and Max-Rate schedulers and a SAC-MLP, all running over the same learned rApp. All schedulers reach a comparable aggregate QoS utility (0.695–0.724), since the window-averaged eMBB rate exceeds $R_{\text{target}}$, so the policies separate in the per-UE satisfaction metrics instead. The D-SAC achieves the highest eMBB SLA satisfaction (0.463, versus 0.429 for PF, 0.403 for Max-Rate, and 0.396/0.394 for Equal/SAC-MLP) and the highest URLLC on-time delivery (0.682, versus 0.649 for Equal and 0.524/0.481 for PF/Max-Rate). The D-SAC leads the SAC-MLP on both metrics; the URLLC margin is narrow (0.682 versus 0.640), but the decisive separation is on eMBB satisfaction, where the flat network collapses to near the Equal scheduler. This is the expected behavior of an index-tied parameterization: because per-UE traffic are randomized each episode, the UE index has no meaning, so without weight sharing the flat actor hedges toward a uniform split, which the Deepset model avoids.




(a) Throughput (Mbps) (b) Jain's fairness

Fig. 4: Throughput and Jain's fairness comparison across scheduling methods.




(a) eMBB SLA satisfaction (b) URLLC on-time delivery

Fig. 5: Scalability performance. D-SAC leads on both.

*3) Service Tradeoff:* Fig. 4 shows the throughput and fairness that each scheduler pays for the service quality. Max-Rate greedily serves users with the best instantaneous channel, reaching the highest aggregate throughput at the cost of the worst URLLC on-time delivery and the lowest fairness, while Equal is the most fair but throughput-limited and leaves much eMBB demand unmet. The D-SAC deliberately trades rate fairness for per-UE SLA satisfaction, which is the right trade when the per-UE minimum-rate contracts are heterogeneous.

*4) Scalability and Robustness to the Number of Users:* Fig. 5 shows that across $30/40/50$ UEs the D-SAC keeps the highest eMBB SLA satisfaction ($0.46$, $0.44$, $0.39$) and the highest URLLC on-time delivery ($0.68$, $0.58$, $0.55$) among all schedulers. The relevant result is that D-SAC retains the lead at every load without retraining: the actor is permutation-equivariant and parameter-shared across UEs, so a single trained model transfers to $40$ and $50$ UEs, whereas the flat SAC-MLP is tied to a fixed UE dimension and must be retrained separately at each count.

## VIII. Conclusion

We proposed a hierarchical intelligent O-RAN controller for a tethered mmWave UAV-gNB serving mixed eMBB and URLLC traffic, in which a non-RT rApp sets the UAV placement and the slice PRB budget as a slow policy and a near-RT xApp performs per-user PRB allocation conditioned on that policy. Realizing the xApp as a permutation-equivariant DeepSets SAC scheduler gives it the natural symmetry of scheduling, unlike a flat MLP, and without the graph construction that GNN-based schedulers require. Evaluated on a ray-traced mmWave environment, the D-SAC improves eMBB SLA satisfaction by up to $17\%$ and URLLC on-time delivery by up to $42\%$ over all classical and learned schedulers, and is the only scheduler high on both service axes. The rApp improves URLLC on-time delivery by up to $20\%$ over random placement and surpasses a strong greedy-coverage heuristic, and the scheduler retains its lead from $30$ to $50$ users without retraining. Overall, the results show the benefit of jointly designing the two RIC control loops around per-UE objectives.

## References


[1] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: Opportunities and challenges," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 36–42, May 2016.

[2] S. Khemiri, M. A. Kishk, and M.-S. Alouini, "Coverage analysis of tethered UAV-assisted large-scale cellular networks," 2023, arXiv:2304.05283.

[3] O-RAN Alliance, "O-RAN architecture description," O-RAN Alliance, Technical Report O-RAN.WG1.O-RAN-Architecture-Description, Feb. 2023, v07.00.

[4] ——, "Deployments of O-RAN-based non-terrestrial networks," O-RAN Alliance, White Paper, Apr. 2025, v08.4.

[5] H. Zou *et al.*, "Telecom world models: Unifying digital twins, foundation models, and predictive planning for 6g," 2026. [Online]. Available: https://arxiv.org/abs/2604.06882

[6] P. Yan, J. Lu, H. Zeng, and Y. T. Hou, "Near-real-time resource slicing for QoS optimization in 5G O-RAN using deep reinforcement learning," *IEEE/ACM Transactions on Networking*, vol. 34, 2026.

[7] M. Zaheer *et al.*, "Deep sets," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[8] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Machine Learning (ICML)*, 2018.

[9] J. Hoydis, F. A. Aoudia, S. Cammerer, F. Vieira, A. Vem, S. ten Brink, and A. Kountouris, "Sionna RT: Differentiable ray tracing for radio propagation modeling," in *Proc. IEEE GLOBECOM Workshops (GC Wkshps)*, Kuala Lumpur, Malaysia, Dec. 2023.

[10] J. Sabzehali, V. K. Shah, H. S. Dhillon, and J. H. Reed, "3D placement and orientation of mmwave-based UAVs for guaranteed LoS coverage," 2021, arXiv:2104.12993.

[11] Y. Wang and J. Farooq, "Deep reinforcement learning based placement for integrated access backhauling in UAV-assisted wireless networks," 2023, arXiv:2312.14247.

[12] P. Luong, F. Gagnon, L.-N. Tran, and F. Labeau, "Deep reinforcement learning-based resource allocation in cooperative UAV-assisted wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 11, pp. 7610–7625, 2021.

[13] S. Lim, H. Yu, and H. Lee, "Optimal tethered-UAV deployment in A2G communication networks: Multi-agent Q-learning approach," *IEEE Internet of Things Journal*, vol. 9, no. 19, pp. 18 539–18 549, 2022.

[14] C. Pham *et al.*, "When RAN intelligent controller in O-RAN meets multi-UAV enabled wireless network," *IEEE Transactions on Cloud Computing*, vol. 11, no. 3, pp. 2245–2259, 2023.

[15] H. Li *et al.*, "Energy-efficient deployment and resource allocation for O-RAN-enabled UAV-assisted communication," *IEEE Transactions on Green Communications and Networking*, vol. 8, no. 3, 2024.

[16] A. E. Giannopoulos, S. T. Spantideas, and P. Trakadas, "CollabORAN: A collaborative rApp–xApp–dApp control architecture for fairness-adaptive resource sharing in O-RAN," 2026, arXiv:2603.20805.

[17] E. Takahashi, T. Onishi, Y. Nishikawa, and K. Date, "Dynamic optimization of per-UE RAN parameters based on latency tolerance via deep learning," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*, 2026.

[18] R. Barker *et al.*, "Real: Reinforcement learning-enabled xapps for experimental closed-loop optimization in o-ran with osc ric and srsran," in *2025 IEEE International Conference on Communications Workshops*, 2025, pp. 389–395.

[19] F. Lotfi and F. Afghah, "Meta reinforcement learning approach for adaptive resource optimization in o-ran," in *2025 IEEE Wireless Communications and Networking Conference (WCNC)*, 2025, pp. 1–6.

[20] O. Semiari, H. Nikopour, and S. Talwar, "Graph reinforcement learning for qos-aware load balancing in open radio access networks," in *2025 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2025, pp. 1948–1953.

[21] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artificial Intelligence (AAAI)*, Phoenix, AZ, USA, Feb. 2016, pp. 2094–2100.

[22] R. K. Jain *et al.*, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," Digital Equipment Corp,, Tech. Rep. DEC-TR-301, 1984.