Mobility-Aware Resource Provisioning for Edge-Assisted Extended Reality Services

Yingying Pei, *Graduate Student Member, IEEE*, Mingcheng He, *Member, IEEE*, Shisheng Hu, *Member, IEEE*, Xinyu Huang, *Member, IEEE*, Conghao Zhou, *Member, IEEE*, Weihua Zhuang, *Fellow, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*—In this article, we propose a novel mobility-aware resource provisioning scheme for edge-assisted extended reality (XR) services. The goal is to minimize resource consumption while satisfying user quality of experience (QoE) requirement, which is measured by the weighted sum of visual quality, quality variation, and round-trip interaction latency. Specifically, we present a mobility model to capture both user spatial movements and XR content interaction features. Since user viewing distance and interaction time are key model parameters that affect the spatiotemporal service demand for XR content rendering and delivery at the edge, we estimate user-specific model parameters and adopt a sample average approximation (SAA) method to model the relationship between user QoE and the consumption of both communication and edge computing resources. We design a coordinate descent algorithm to make resource provisioning decisions, where a deep neural network (DNN) provides a valuable initial point to accelerate convergence. Simulation results demonstrate that our proposed scheme is more efficient to utilize network resources in comparison with benchmark schemes while satisfying user QoE requirements.**

***Index Terms*—Extended reality (XR), mobility model, quality of experience (QoE), resource provisioning.**

I. Introduction

Extended reality (XR) refers to a collection of immersive technologies, including augmented reality (AR), mixed reality (MR), virtual reality (VR), and everything in between [1]. These technologies rely on XR devices, such as AR glasses and VR headsets, to enable real-time interaction with virtual content. For example, in an AR museum, visitors wearing AR headsets can explore virtual exhibits as if they were physically present. Such immersive experiences are realized by real-time XR content rendering throughout the viewing process [2], [3]. Specifically, XR content, such as volumetric video, is converted into video frames based on the user's pose (i.e., position and orientation) and then displayed on the XR device. To ensure an immersive user experience, both low latency and high-quality rendering are essential [4], [5].

Standalone XR devices are typically lightweight and have limited computing capabilities, restricting their ability to perform both timely and high-quality rendering locally. To address this limitation, mobile edge computing offers a promising solution by providing computing resources on edge servers for XR content rendering [6], [7]. Specifically, XR devices can offload computationally intensive rendering tasks to nearby edge servers, which then efficiently perform the rendering and deliver the rendered video frames back to the devices. In this way, XR devices can achieve high-quality and low-latency rendering, thus enhancing the user quality of experience (QoE).

To consistently guarantee user QoE, a common approach is to proactively provision dedicated network resources, including communication and computing resources, for XR services [8]. However, efficient resource provisioning remains challenging due to the inherent uncertainty and dynamic nature of future service demands [9], [10]. To address this challenge, existing approaches for conventional services typically rely on stochastic modeling of spatiotemporal service demands to enable on-demand resource provisioning. Specifically, temporal service demand features, such as the arrival patterns of service requests, are commonly modeled using various traffic models, such as the Poisson process [11] and the on–off process [12]. For user spatial movement modeling, mobility models such as the random waypoint (RWP) model for mobile users [13] and the fluid-flow model for vehicles [14] are widely adopted. By combining mobility and traffic models, the spatiotemporal service demand can be effectively characterized, enabling efficient resource provisioning.

In addition to the existing studies, further research is needed to accurately capture the spatiotemporal demand dynamics of XR services. The reasons are as follows. First, unlike traditional services, XR users exhibit distinct movement patterns, typically alternating between two phases: a *travel* phase and an *interaction* phase. During the travel phase, users navigate between different virtual content and generate only minimal rendering demand (e.g., directional cues such as arrows). In the interaction phase, users engage with specific virtual content (e.g., manipulating virtual exhibits), which requires frequent and intensive rendering. Characterizing the relative durations of the travel and interaction phases is essential for

Received 2 August 2025; revised 7 October 2025; accepted 1 November 2025. Date of publication 17 November 2025; date of current version 8 January 2026. An earlier version of this paper was presented in part at the IEEE/CIC International Conference on Communications in China, Shanghai, China, 2025 [DOI: 10.1109/ICCC65529.2025.11149304]. *(Corresponding author: Shisheng Hu.)*

Yingying Pei, Mingcheng He, Shisheng Hu, Xinyu Huang, Weihua Zhuang, and Xuemin Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: y32pei@uwaterloo.ca; mingcheng.he@uwaterloo.ca; s97hu@uwaterloo.ca; x357huan@uwaterloo.ca; wzhuang@uwaterloo.ca; sshen@uwaterloo.ca).

Conghao Zhou is with the School of Telecommunications Engineering, Xidian University, Xi'an 710071, China (e-mail: zhouconghao@xidian.edu.cn).

Digital Object Identifier 10.1109/JIOT.2025.3633587

2327-4662 © 2025 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

estimating the overall service demand for XR. Second, during the interaction phase, the service demand is further influenced by the viewing distance between the XR user and the virtual content [15]. Specifically, as viewing distance increases, the virtual content appears smaller in the user's viewport, thus reducing perceptual sensitivity to rendering quality. As a result, high-quality rendering (e.g., high resolution or high frame rate) becomes unnecessary for distant virtual content. Therefore, capturing users' preferred viewing distances is also crucial for accurate service demand estimation.

In this article, we propose an efficient resource provisioning scheme for edge-assisted XR services by explicitly considering the unique mobility characteristics of XR users. In particular, we present a mobility model that incorporates XR-specific interaction features to characterize users' spatiotemporal service demands. Based on this model, we extract user-specific model parameters from historical viewing trajectories. The parameterized model serves two purposes: 1) it generates a large number of trajectory samples for sample average approximation (SAA), enabling statistical analysis of the relationship between QoE and resource consumption; and 2) it is used as input, along with XR environmental parameters, to a deep neural network (DNN) for predicting future resource demand. Then, we design a low-complexity coordinate descent algorithm to optimize resource provisioning, which minimizes the weighted cost of communication and computing resources while ensuring QoE satisfaction. The algorithm leverages the DNN output as a high-quality initial point, significantly improving the convergence speed of the algorithm. The main contributions are summarized as follows.

1) We propose a mobility-aware resource provisioning scheme for edge-assisted XR services by considering key user mobility features, including viewing distance and interaction time, enabling accurate estimation of the spatiotemporal service demand and efficient resource utilization.
2) We present a mobility model tailored for XR users that use user-specific mobility parameters to enable both trajectory sampling-based QoE estimation and DNN-based resource demand prediction under user dynamics.
3) We design a low-complexity coordinate descent algorithm to minimize the weighted cost of communication and computing resources while ensuring user QoE satisfaction. The algorithm uses DNN outputs as initialization to accelerate convergence.

The remainder of this article is organized as follows. Section II reviews the related work. Section III introduces the system model and formulates a resource provisioning problem. Section IV presents the proposed mobility-aware resource provisioning scheme. Simulation results are presented in Section V, and Section VI concludes this article.

II. Related Work

Efficient resource provisioning aims to minimize network resource consumption while satisfying the dynamic and uncertain service demands of mobile users [9]. Achieving this goal requires accurately modeling users' spatiotemporal service demands and establishing effective mappings from these demands to the required network resources. Existing research in this area can be broadly classified into three categories: model-based, data-driven, and hybrid methods [16].

Model-based methods utilize mathematical models and statistical information to characterize the spatiotemporal service demand, typically by integrating traffic and user mobility models. Specifically, traffic models describe when and how much data are generated or requested, while mobility models characterize where and when users or devices are present within the network. For traditional services, widely adopted traffic models include the Poisson process and the two-state Markov (on–off) model [12]. In the context of emerging XR services, traffic modeling often incorporates application-level features (e.g., encoding modes [17], [18]) or user viewing behaviors [19], [20]. For example, a semi-hidden Markov model was used in [19] to characterize XR traffic patterns generated by point cloud encoding. In terms of user mobility, models such as the RWP model and the fluid-flow model are commonly used to represent the movement patterns of mobile users and high-speed moving vehicles, respectively. Based on these traffic and mobility models, analytical tools such as capacity theory [21] and probabilistic analysis [22] are used to provide statistical guarantees for quality of service (QoS) or user QoE while optimizing resource utilization.

Data-driven methods leverage advanced neural networks to model the complex relationships between user behavior and multidimensional resource provisioning, without the need for explicit mathematical formulations [23], [24], [25]. In addition, the incorporation of prediction algorithms enables proactive analysis of user mobility and future capacity demands, facilitating efficient resource provisioning across network cells and edge nodes [26], [27], [28], [29]. Hybrid methods combine the advantages of both model-based and data-driven methods to enable enhanced robustness and adaptability in dynamic network environments [30]. For example, mathematical models are first used to characterize dynamic service demands, while data-driven methods are then utilized to evaluate and refine these models [31].

Benefiting from the interpretability and analyzability of model-based methods, we present a mobility model tailored for emerging XR services. By incorporating the unique movement and interactive features of XR users, the presented model can effectively characterize spatiotemporal service demands, thus facilitating efficient resource provisioning.

III. System Model and Problem Formulation

A. Network Scenario

As shown in Fig. 1, we consider an edge-assisted XR system consisting of an access point (AP), an edge server co-located with the AP, multiple XR users, and a set of virtual objects. Users equipped with XR devices, denoted by set $\mathcal{I} = \{1, \ldots, I\}$, can view and interact with virtual objects, such as digital exhibits in an AR museum. Virtual objects, denoted by set $\mathcal{Z} = \{1, \ldots, Z\}$, are stored on the edge server as static or dynamic point clouds. Each object supports L discrete density levels, denoted by set $\mathcal{L} = \{1, \ldots, L\}$. Each density level corresponds to a specific point cloud resolution, with

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

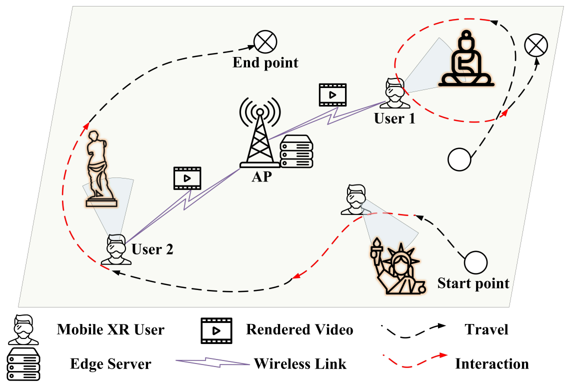


Fig. 1. Illustration of the edge-assisted XR system.

higher levels preserving finer geometric details for close-up inspection [32].

To maintain a seamless immersive experience, each XR device continuously tracks its pose using computer vision techniques (e.g., simultaneous localization and mapping). When the pose change exceeds a predefined threshold, the device uploads the updated pose to the edge server as a rendering request. Based on the pose information, the server first checks the user's view frustum to identify which virtual objects fall within the user's field of view [33]. It then selects an appropriate density level and executes the rendering pipeline (including shading, lighting, and rasterization) to generate pose-aligned video frames. We assume that only the virtual object currently within the user's field of view and closest to the viewport is rendered, such as in the case that a visitor focuses on one exhibit at a time. Finally, the rendered frames are transmitted back to the XR device over wireless links for display [34].

Due to the dynamic nature of user movement and interaction, resource demands fluctuate over time. To continuously guarantee user QoE while improving network resource utilization, we adopt a two-timescale resource management framework to support XR services [9], [21], [22]. Specifically, at the large timescale, radio spectrum bandwidth and computing resources are provisioned over a predefined planning window (e.g., 30 min). This provisioning ensures a minimum level of resources for XR services, even under significant traffic fluctuations from other services. At the small timescale, the provisioned resources are dynamically allocated among XR devices, and rendering quality levels are adaptively selected to maximize overall user QoE [35]. In this work, we focus on designing an efficient large-timescale resource provisioning scheme under a predefined small-timescale resource scheduling policy.

Let a and b denote the amounts of bandwidth (in Hz) and computing resources [in floating-point operations per second (FLOPS)], respectively, provisioned for the entire planning window.[1] The planning window is uniformly divided into T consecutive scheduling slots, denoted by set $\mathcal{T} = \{1, \ldots, T\}$. For each slot $t \in \mathcal{T}$, we define three vectors: radio resource allocation vector $\boldsymbol{\alpha}_t = [\alpha_{1,t}, \ldots, \alpha_{I,t}]^\top$, computing resource allocation vector $\boldsymbol{\beta}_t = [\beta_{1,t}, \ldots, \beta_{I,t}]^\top$, and rendering quality selection vector $\boldsymbol{\gamma}_t = [\gamma_{1,t}, \ldots, \gamma_{I,t}]^\top$. Here, $\alpha_{i,t}$ and $\beta_{i,t}$ denote the radio and computing resources allocated to device $i \in \mathcal{I}$, while $\gamma_{i,t} \in \mathcal{L}$ represents the rendering quality level selected for device $i \in \mathcal{I}$. The per-slot scheduling decisions are determined by a predefined mapping policy, denoted by $\pi(\cdot)$, given by

$$\left\{\boldsymbol{\alpha}_t, \boldsymbol{\beta}_t, \boldsymbol{\gamma}_t\right\} = \pi\left(a, b, \mathbf{U}_t\right) \tag{1}$$

where $\mathbf{U}_t = \{\boldsymbol{u}_{i,t}\}_{i=1}^{I}$ denotes the real-time system state at slot t. Each $\boldsymbol{u}_{i,t} = \{z_{i,t}, d_{i,t}, h_{i,t}\}$ represents the state of device i. Specifically, the following conditions hold.

1) $z_{i,t} \in \mathcal{Z}$ is the index of the currently viewed object, which affects the computational complexity of the rendering task due to the object's intrinsic geometric complexity.
2) $d_{i,t}$ is the viewing distance, which refers to the spatial distance between the user's viewpoint and the virtual object being viewed. It influences the required rendering quality level.
3) $h_{i,t}$ is the instantaneous channel condition, which determines the efficiency of radio resource usage during the transmission of rendered video frames.

The first two parameters can be inferred from the users' real-time pose data and the spatial layout of virtual objects [20], while the third parameter can be estimated through techniques such as pilot-based signal measurements [37].

B. Round-Trip Interaction Latency Analysis

According to [2] and [3], ensuring satisfactory round-trip interaction latency, recognized as a typical QoS metric, is essential for delivering immersive XR experiences. Excessive latency increases the mismatch between displayed virtual content and user expectations, which in turn reduces user immersion [38]. Round-trip interaction latency is defined as the time from when a user's pose is changed to the moment when the corresponding rendered video frame is displayed on the XR device. Specifically, this latency consists of the following components.

1) Pose Estimation Latency: This component refers to the time required by the XR device to collect and process pose data (e.g., head pose tracking). We model this latency as a constant τ_0 due to the availability of dedicated local resources.[2] In typical XR systems, τ_0 can be maintained within 20 ms with submillimeter accuracy [39].

2) Rendering Latency: After pose estimation, the XR device transmits the pose data to the edge server. Given that pose data are relatively small compared to video content, the uplink transmission latency is considered negligible. Upon receiving the pose data, the edge server executes the rendering process. Let $\tau_{i,t}^{\text{ren}}$ denote the rendering latency experienced by user i at slot t, given by

$$\tau_{i,t}^{\text{ren}} = \frac{\rho^{\text{ren}} \psi\left(z_{i,t}, \gamma_{i,t}\right)}{\beta_{i,t}} \tag{2}$$

[1]Although a complete rendering pipeline consumes various types of computing resources (e.g., GPU, CPU, and memory), we focus only on the GPU FLOPS and assume that each allocated FLOPS unit is provisioned together with a proportional share of other computing resources [36].

[2]The latency between the actual pose change and its detection by sensors (e.g., cameras) is ignored, as it is negligible compared to the round-trip interaction latency.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

where $\psi(z_{i,t}, \gamma_{i,t})$ (in bits) represents the size of the rendering data, which depends on the viewed virtual object $z_{i,t}$ and selected rendering quality level $\gamma_{i,t}$. In (2), ρ^{ren} denotes the computational intensity (in FLOP per bit) required for point cloud rendering.

3) Transmission Latency: After rendering, a video frame of size $\chi(\gamma_{i,t})$ (in bits) is generated at the edge server and transmitted to the XR device.[3] Let $\tau^{\text{tx}}_{i,t}$ denote the downlink transmission latency for user i at slot t, given by

$$\tau^{\text{tx}}_{i,t} = \frac{\chi\left(\gamma_{i,t}\right)}{\alpha_{i,t}\log_2\left(1+\frac{P_0 h_{i,t}}{\sigma^2}\right)} \tag{3}$$

where P_0 denotes the transmission power, $h_{i,t}$ is the downlink channel gain between the AP and device i at slot t, and σ^2 is the average received noise power.

4) Round-Trip Interaction Latency: Let $\tau^{\text{rt}}_{i,t}$ denote the round-trip interaction latency experienced by user i at slot t, given by

$$\tau^{\text{rt}}_{i,t} = \tau_0 + \tau^{\text{ren}}_{i,t} + \tau^{\text{tx}}_{i,t}. \tag{4}$$

C. QoE Model

To evaluate user QoE, we jointly consider three factors that affect a user's perceived immersion and interaction fluency in XR applications: visual quality, quality variation, and the round-trip interaction latency derived in Section III-B.

1) Visual Quality: The visual quality depends on both the rendering quality and the user's viewing distance [15] and is modeled as

$$V_{i,t} = \zeta\left(d_{i,t}\right)\gamma_{i,t} \tag{5}$$

where $\zeta(d_{i,t})$ is a nonnegative weight function that quantifies the user's sensitivity to visual quality as a function of viewing distance $d_{i,t}$. This sensitivity decreases as the viewing distance increases.

2) Quality Variation: Temporal fluctuations in visual quality can disrupt user immersion. The quality variation between two successive video segments is given by

$$\Delta V_{i,t} = \left|V_{i,t} - V_{i,t-1}\right| \tag{6}$$

where a larger $\Delta V_{i,t}$ indicates more severe visual fluctuations, which negatively affect user QoE [40].

3) Round-Trip Latency: To characterize the impact of round-trip latency on user QoE, we use a sigmoid-based utility function [41], [42]. Let $\hat{\tau}$ denote the latency threshold, representing the maximum tolerable round-trip latency for maintaining seamless immersion. The latency utility function is given by

$$\mathrm{U}\left(\tau^{\text{rt}}_{i,t}\right) = \frac{1}{1+\exp\left[\upsilon\left(\tau^{\text{rt}}_{i,t}-\hat{\tau}\right)\right]} \tag{7}$$

where υ is a constant parameter to control the steepness of QoE degradation when the round-trip latency exceeds the threshold. In particular, (7) reflects that, when $\tau^{\text{rt}}_{i,t} \ll \hat{\tau}$, the utility is almost insensitive to any change in latency. However, as the latency approaches or exceeds the threshold $\hat{\tau}$, the utility begins to decrease sharply with further increases in $\tau^{\text{rt}}_{i,t}$.

[3]We assume that frames rendered at the same quality level have identical resolution and data size. Hence, the object index is omitted.

4) User QoE: The user QoE is modeled as a linear combination of the above three factors. For user i at slot t, the QoE is given by

$$Q_{i,t} = \mu_1 V_{i,t} + \mu_2 \Delta V_{i,t} + \mu_3 U\left(\tau^{\text{rt}}_{i,t}\right) \tag{8}$$

where $\mu_1 > 0$, $\mu_2 < 0$, and $\mu_3 > 0$ are weighting factors that reflect the relative importance of visual quality, quality variation, and latency utility in the user QoE.

D. Problem Formulation

At the beginning of each planning window, a resource provisioning decision is made and remains fixed throughout the window. Our objective is to minimize the weighted cost of radio and computing resources while ensuring that the expected QoE meets a minimum requirement, $Q_{\min}$. The expected QoE across all users and time slots, denoted by $\mathbb{E}[\bar{Q}]$, is given by

$$\mathbb{E}\left[\bar{Q}\right] = \frac{1}{TI}\sum_{t=1}^{T}\sum_{i=1}^{I}\mathbb{E}_{\tilde{\mathbf{U}}_t}\left[Q_{i,t}(a,b)\right] \tag{9}$$

where $\tilde{\mathbf{U}}_t$ represents the uncertain system state at slot t, modeled as a random variable and unknown at the beginning of the planning window. The term $\mathbb{E}_{\tilde{\mathbf{U}}_t}[Q_{i,t}(a,b)]$ denotes the expected QoE of user i at slot t, conditioned on provisioned resource tuple (a,b) and system state $\tilde{\mathbf{U}}_t$. The resource provisioning problem is then formulated as

$$\mathbf{P1}: \min_{a,b} \quad \omega_a a + \omega_b b \tag{10a}$$

$$\text{s.t.}\ \mathbb{E}\left[\bar{Q}\right] \geq Q_{\min} \tag{10b}$$

$$a \in [0, A_{\max}], \quad b \in [0, B_{\max}] \tag{10c}$$

where ω_a and ω_b are positive weighting coefficients representing the costs of radio and computing resources, respectively. Constraint (10b) ensures that the expected QoE meets the required threshold, while constraint (10c) specifies the maximum resource capacities, where $A_{\max}$ and $B_{\max}$ denote the capacity of radio and computing resources.

Solving Problem **P1** requires accurately capturing the relationship between the provisioned resources and the expected QoE. A major challenge lies in the stochastic and dynamic nature of system state $\tilde{\mathbf{U}}_t$, which is fundamentally driven by user mobility in XR environments. As users move and interact with spatially distributed virtual content, their service demands vary over time and space, leading to nonstationary and uncertain resource-QoE relationships. Therefore, an accurate characterization of user mobility features is critical for guiding efficient resource provisioning. To address this challenge, we propose a mobility-aware resource provisioning scheme that leverages user-specific mobility parameters to effectively capture spatiotemporal variations in service demands.

IV. Resource Provisioning Scheme

In this section, we present a resource provisioning scheme to solve Problem **P1**. We first give an overview of the proposed scheme, followed by detailed descriptions of its key components.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

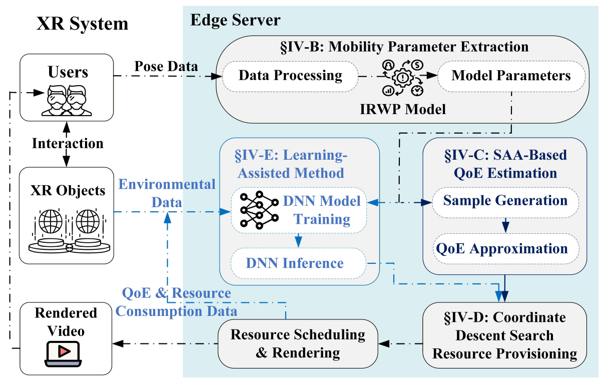

XR System
Edge Server
Users
Pose Data
§IV-B: Mobility Parameter Extraction
Data Processing
Model Parameters
IRWP Model
Interaction
XR Objects
Environmental Data
§IV-E: Learning-Assisted Method
DNN Model Training
DNN Inference
§IV-C: SAA-Based QoE Estimation
Sample Generation
QoE Approximation
QoE & Resource Consumption Data
Rendered Video
Resource Scheduling & Rendering
§IV-D: Coordinate Descent Search Resource Provisioning

Fig. 2. Procedure of the resource provisioning scheme.

A. Scheme Overview

Fig. 2 illustrates the procedure of the proposed scheme. As XR users navigate and interact with virtual objects, their pose data are continuously uploaded to the edge server. These data are processed by a mobility analysis module to extract user-specific parameters. These parameters serve two purposes. First, they are used to generate synthetic trajectories to approximate future system states. Based on these trajectories, the expected QoE under any candidate resource provisioning pair is estimated using the SAA method (shown in dark blue). Second, the extracted mobility parameters are combined with real-time resource consumption data, instantaneous QoE measurements, and XR environmental parameters to train a lightweight DNN (shown in light blue). Finally, a coordinate descent search-based algorithm is used to determine the resource provisioning decision, where the DNN output serves as an initial point to accelerate convergence.

B. Mobility Parameter Extraction

Based on the RWP mobility model, we present an approach to extracting user mobility parameters that can be used to efficiently solve **P1**. The RWP model is a widely used mobility model in wireless networks to simulate the movement of mobile users [43]. It characterizes user movement between randomly selected waypoints and the pause time at each waypoint. To better support XR scenarios, we present an interactive RWP (IRWP) model customized to XR users that incorporates viewing behaviors during user pauses at waypoints.

1) Mobility Parameters: Following the modeling approach in [43], we assume that all users move independently and have identical stochastic mobility properties. In the following, we focus on the movement of a single user. Specifically, the movement of each user is characterized by a sequence of discrete movement periods indexed by $n \in \mathbb{N}$, each consisting of two phases.

1) *Travel Phase:* This phase inherits user movement defined in the RWP model, which is the transition from previous waypoint D_{n-1} to new waypoint D_n at moving velocity v_n.
2) *Interaction Phase:* During the pause at D_n, the user keeps changing its viewing distance, represented by a sequence of discrete viewing states, $\boldsymbol{s}_n = [s_{n,1}, s_{n,2}, \ldots, s_{n,O_n}]$, where O_n denotes the pause time at D_n, measured in time slots.

The IRWP model is defined by a sequence of tuples $\{D_n, v_n, \boldsymbol{s}_n\}_{n\in\mathbb{N}}$, where $O_n = |\boldsymbol{s}_n|$, and an additional waypoint, D_0, is needed for initialization. An illustration of the IRWP model is given in Fig. 3.

2) Parameter Extraction: The IRWP model parameters, D_n, v_n, and O_n, can be determined in the same manner as in the classical RWP model [43]. In the following, we focus on the determination of viewing state sequence parameter $\boldsymbol{s}_n$. We model changes in viewing distance using a Markov chain, where the transition probabilities between discrete viewing distance states are encoded in a Markov transition matrix. Assume that the state transition probabilities remain constant within a planning window but may vary across different planning windows.

Let $\mathcal{G} = \{1, \ldots, G\}$ denote the set of all possible discrete viewing states. For user i at time slot t, the continuous viewing distance, $d_{i,t}$, is quantized into a discrete state, $\phi_{i,t} \in \mathcal{G}$, according to

$$\phi_{i,t} = \begin{cases} \left\lfloor \dfrac{d_{i,t}}{\Delta d} \right\rfloor + 1, & \text{if } d_{i,t} \le d_{\max} \\ G, & \text{otherwise} \end{cases} \tag{11}$$

where Δd denotes the distance interval between consecutive discrete states, $d_{\max}$ denotes the maximum threshold for distance quantization, and $\lfloor\cdot\rfloor$ is the floor function. For user i currently in state $g_1 \in \mathcal{G}$, the probability of transitioning to state $g_2 \in \mathcal{G}$ in the next time slot is denoted by $\Pr(\phi_{i,t+1} = g_2 \mid \phi_{i,t} = g_1)$. This transition probability is estimated based on the empirical frequency of observed state transitions during the previous planning window, given by

$$\begin{aligned} &\Pr\left(\phi_{i,t+1} = g_2 \mid \phi_{i,t} = g_1\right) \\ &\quad = \frac{\sum_{t=1}^{T-1} I\left(\phi_{i,t+1} = g_2 \wedge \phi_{i,t} = g_1\right)}{\sum_{t=1}^{T-1} I\left(\phi_{i,t} = g_1\right) + \epsilon} \end{aligned} \tag{12}$$

where $I(\cdot)$ denotes the indicator function, which equals one if the specified condition is satisfied and zero otherwise; and ϵ is a small positive constant added to avoid division by zero when state g_1 has never been visited. Note that state G is defined as the condition in which the viewing distance exceeds $d_{\max}$. While such distances often occur during the travel phase, the travel phase itself is not considered a state in the Markov process.

C. SAA-Based QoE Estimation

Given the extracted mobility parameters, we present an efficient approach to estimate the expected QoE in **P1** using SAA [44]. Instead of directly calculating the expectation of a random objective function or constraint, SAA approximates the expected value by replacing it with an empirical average over a finite set of randomly generated samples.

1) Scenario Construction: Let Ω_i denote the set of all possible viewing trajectory realizations for user i. A system-level scenario is constructed by randomly sampling a trajectory realization $\omega_i \in \Omega_i$ for each user and combining these into a joint realization. Specifically, a scenario is denoted as

$$D_0 \xrightarrow{\nu_1} \underbrace{\boxed{[s_{1,1}, s_{1,2}, \ldots, s_{1,O_1}]}}_{\text{Pausing at } D_1} \xrightarrow{\nu_2} \underbrace{\boxed{[s_{2,1}, s_{2,2}, \ldots, s_{2,O_2}]}}_{\text{Pausing at } D_2} \longrightarrow \cdots \longrightarrow \underbrace{\boxed{[s_{n,1}, s_{n,2}, \ldots, s_{n,O_n}]}}_{\text{Pausing at } D_n} \xrightarrow{\nu_n} \cdots$$

Fig. 3. Illustration of the IRWP model.

$\xi = \{(\omega_1, \ldots, \omega_I) \mid \omega_i \in \Omega_i, \forall i \in \mathcal{I}\}$, representing one possible realization of the overall system state throughout the planning window.

2) QoE Approximation: Given scenario ξ_j, predefined scheduling policy π, and resource provisioning tuple (a, b), the user QoE over the planning window, denoted by $Q(a, b, \xi_j)$, can be calculated using (2)–(8). The SAA-based approximation of the expected QoE is then given by

$$\tilde{Q}_N(a, b) = \frac{1}{N}\sum_{j=1}^{N} Q\left(a, b, \xi_j\right) \tag{13}$$

where $\tilde{Q}_N(a, b)$ serves as the empirical estimate of the expected QoE, i.e., $\mathbb{E}[\bar{Q}]$ in constraint (10b), based on N sampled scenarios. The value of N is predetermined to balance the tradeoff between estimation accuracy and computational complexity. The detailed procedures to estimate QoE are given in Algorithm 1.

```
Algorithm 1 SAA-Based QoE Estimation
Input: Resource provisioning tuple $(a, b)$, IRWP model parameters;
1 Initialization: For each user $i$, generate a sufficiently large set of trajectory samples $\Omega_i$ using the IRWP model;
2 for $j = 1$ to $N$ do
3     Construct scenario $\xi_j$ by randomly selecting one trajectory sample for each user $\xi_j = \{\omega_i \in \Omega_i \mid i \in \mathcal{I}\}$;
4     Simulate scheduling policy $\pi$ over $T$ time slots to determine scheduling decisions;
5     Compute QoE $Q(a, b, \xi_j)$ for scenario $\xi_j$ according to (2)–(8);
6 end for
7 Calculate sample-averaged QoE $\tilde{Q}_N(a, b)$ according to (13).
Output: Estimated QoE $\tilde{Q}_N(a, b)$.
```

3) Complexity Analysis: The computational complexity of Algorithm 1 can be estimated as follows. In the scenario construction phase, there are I users, each generating $|\Omega_i|$ viewing trajectories over T time slots. At each slot, the next viewing state is sampled from a Markov process with G possible states, which requires $O(G)$ operations per transition. Therefore, the complexity of scenario construction is at most $O(\mathrm{GIT}\Omega_{\max})$, where $\Omega_{\max} = \max_{i\in\mathcal{I}} |\Omega_i|$ denotes the maximum number of trajectory samples across all users. In the QoE-estimation phase, N sample scenarios are evaluated. Within each scenario and each time slot, the scheduler generates a resource allocation decision via policy π, followed by the computation of instantaneous QoE for each of the I users. Let H denote the number of operations required to compute the resource allocation and QoE for a single user in a single slot. The total complexity of the QoE estimation phase is therefore $O(\mathrm{NITH})$. Combining both phases, the overall computational complexity of Algorithm 1 is $O(\mathrm{GIT}\Omega_{\max} + \mathrm{NITH})$.

D. Coordinate Descent Search

By replacing $\mathbb{E}[\bar{Q}]$ with its approximation in (13), Problem **P1** can be reformulated as

$$\textbf{P2}: \min_{a,b} \quad \omega_a a + \omega_b b \tag{14a}$$
$$\text{s.t. } Q_{\min} - \tilde{Q}_N(a, b) \le 0 \tag{14b}$$
$$(10c).$$

Problem **P2** is a constrained two-variable optimization problem. Under certain scheduling policies, such as the proportional-sharing policy (see the details in Appendix A), constraint (14b) is convex with respect to both a and b (see the proof in Appendix B). Therefore, Problem **P2** can be efficiently solved using the coordinate descent method [45].[4] Coordinate descent is an iterative optimization technique in which, at each iteration, a subset of the variables is held fixed, while the objective function is optimized with respect to the remaining variables. Specifically, we alternatively optimize bandwidth and computing resource provisioning decisions.

1) Computing Resource Provisioning: At the mth iteration, given the current bandwidth provisioning, $a^{(m)}$, we solve Problem **P2** to determine the optimal computing resource provisioning, $b^{(m)}$. Since $\tilde{Q}_N(a^{(m)}, b)$ is a nondecreasing function with respect to b, we can use the binary search algorithm [46] to efficiently determine the minimum value of $b^{(m)}$ that satisfies the QoE constraint, i.e.,

$$\left|\tilde{Q}_N\left(a^{(m)}, b^{(m)}\right) - Q_{\min}\right| \le \epsilon_Q \tag{15}$$

where ϵ_Q is a small positive tolerance to ensure convergence. The resulting $b^{(m)}$ corresponds to the optimal solution for computing resource provisioning in the current iteration of Problem **P2**.

2) Bandwidth Resource Provisioning: Given the tightly satisfied constraint $\tilde{Q}_N(a^{(m)}, b^{(m)}) \approx Q_{\min}$, we define an implicit function, $b^{(m)} = \varphi(a^{(m)})$, through the equality condition

$$\tilde{Q}_N\left(a^{(m)}, \varphi\left(a^{(m)}\right)\right) = Q_{\min}. \tag{16}$$

For brevity, we omit the iteration index m in the following derivations. By differentiating (16), we have

$$\frac{\partial \tilde{Q}_N(a, \varphi(a))}{\partial a} + \frac{\partial \tilde{Q}_N(a, \varphi(a))}{\partial b}\varphi'(a) = 0 \tag{17}$$

where $\varphi'(a)$ denotes the derivative of $\varphi(a)$ with respect to a. We further have

$$\varphi'(a) = -\frac{\partial \tilde{Q}_N(a, \varphi(a))}{\partial a}\left[\frac{\partial \tilde{Q}_N(a, \varphi(a))}{\partial b}\right]^{-1}. \tag{18}$$

Next, we define a cost function, $\mathrm{F}(a)$, along the constraint boundary, given by

$$\mathrm{F}(a) = \omega_a a + \omega_b \varphi(a). \tag{19}$$

[4]Note that the convexity is not guaranteed under all scheduling policies. In such cases, the coordinate descent method is treated as a heuristic method.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

Let $\nabla \mathrm{F}(a)$ represent the gradient of the cost function with respect to a, given by

$$\nabla \mathrm{F}(a) = \omega_a - \omega_b \times \frac{\partial \tilde{Q}_N(a, \varphi(a))}{\partial a} \left[\frac{\partial \tilde{Q}_N(a, \varphi(a))}{\partial b} \right]^{-1}. \quad (20)$$

Since the partial derivatives in (20) cannot be computed analytically, we approximate them using finite differences [45]. Specifically, the partial derivatives of $\tilde{Q}_N(a, b)$ with respect to the bandwidth and computing resource values are estimated as

$$\frac{\partial \tilde{Q}_N(a, \varphi(a))}{\partial a} \approx \frac{\tilde{Q}_N(a+\delta, b) - \tilde{Q}_N(a-\delta, b)}{2\delta} \quad (21)$$

$$\frac{\partial \tilde{Q}_N(a, \varphi(a))}{\partial b} \approx \frac{\tilde{Q}_N(a, b+\delta) - \tilde{Q}_N(a, b-\delta)}{2\delta} \quad (22)$$

where δ is a small positive constant used to perturb the current bandwidth or computing resource value for numerical differentiation. Note that the evaluations of $\tilde{Q}_N(a \pm \delta, b)$ and $\tilde{Q}_N(a, b \pm \delta)$ are carried out by executing Algorithm 1.

Given the gradient of the cost function evaluated at the mth iteration, denoted by $\nabla \mathrm{F}(a^{(m)})$, the bandwidth resource is updated via gradient descent

$$a^{(m+1)} = P_A\left(a^{(m)} - \eta^{(m)} \nabla F\left(a^{(m)}\right)\right) \quad (23)$$

where $P_A(x) = \max(0, \min(x, A_{\max}))$ projects the updated value onto feasible bandwidth range $[0, A_{\max}]$ and $\eta^{(m)} > 0$ denotes the step size at the mth iteration. To improve convergence stability near the optimum, we adopt an adaptive step size that decreases with the number of iterations, given by $\eta^{(m)} = \eta_0/(1+m)$, where η_0 is the initial step size. The coordinate descent search algorithm is given in Algorithm 2.

3) Complexity Analysis: In iteration m of Algorithm 2, the computational complexity is dominated by two components: the binary search for computing $b^{(m)}$ (line 3) and the gradient calculation (lines 4–6). The binary search requires approximately $O(\log_2(B_{\max}/\epsilon_Q))$ iterations. In each iteration, the SAA-based QoE estimation is performed once. Therefore, the total complexity of the binary search is $O(\mathrm{NITH}\log_2(B_{\max}/\epsilon_Q))$. The gradient computation involves four SAA-based QoE evaluations for finite difference approximation, incurring a complexity of $O(4\mathrm{NITH})$. Since $\log_2(B_{\max}/\epsilon_Q)$ is typically much larger than 4, the overall computational complexity of Algorithm 2 is given by $O(\mathrm{CNITH}\log_2(B_{\max}/\epsilon_Q))$, where C denotes the number of coordinate descent iterations.

E. Learning-Assisted Method

In this section, we use a supervised learning-based DNN model [47] to learn the mapping from user viewing behavior and XR environment deployment to the required resource consumption under a given QoE requirement. Specifically, at the end of each planning window, we collect relevant system data and retrain the DNN model. Simultaneously, the trained model is used to predict the resource consumption required for the upcoming planning window. The DNN output is then used as the initial input for Algorithm 2. In the following, we describe how the collected data are processed as inputs and outputs for the DNN training and inference.

Algorithm 2 Coordinate Descent Search for Resource Provisioning

```
Input: QoE requirement $Q_{\min}$, convergence tolerances $\epsilon_a, \epsilon_F$,
       cost weights $\omega_a, \omega_b$;
1 Initialization: Set iteration index $m = 0$, initial resource
  provisioning $(a^{(0)}, b^{(0)})$;
2 repeat
3     Given $a^{(m)}$, solve Problem P2 using the binary search
      algorithm to obtain optimal $b^{(m)}$;
4     Compute partial derivatives using (21) and (22), then
      estimate the gradient via (20);
5     Update $a^{(m+1)}$ according to the gradient descent rule in
      (23);
6     Increment iteration index: $m \leftarrow m + 1$;
7 until $|a^{(m)} - a^{(m-1)}| \le \epsilon_a$ and $|\mathrm{F}^{(m)} - \mathrm{F}^{(m-1)}| \le \epsilon_F$
8 Set $a^\star \leftarrow a^{(m)}$, $b^\star \leftarrow b^{(m)}$;
Output: Final resource provisioning decision $(a^\star, b^\star)$.
```

1) Long-Term User Viewing Parameter: Let $\mathcal{K} = \{1, \ldots, K\}$ denote the set of considered planning windows and $\mathcal{T}_k = \{(k-1)T+1, \ldots, kT\}$ denote the set of time slots within planning window k. We define the long-term viewing parameter vector for user i during planning window k as $\boldsymbol{f}_{i,k}^{\mathrm{long}} = [\lambda_{i,k}^{\mathrm{avg}}, d_{i,k}^{\mathrm{avg}}]$, where $\lambda_{i,k}^{\mathrm{avg}}$ denotes the average interaction proportion and $d_{i,k}^{\mathrm{avg}}$ represents the average viewing distance. Average interaction proportion $\lambda_{i,k}^{\mathrm{avg}}$ quantifies the fraction of time slots during which user i is within the valid viewing distance (i.e., $d_{i,t} \le d_{\max}$) for virtual content interaction. It is given by

$$\lambda_{i,k}^{\mathrm{avg}} = \frac{1}{T} \sum_{t \in \mathcal{T}_k} I\left(d_{i,t} \le d_{\max}\right). \quad (24)$$

Average viewing distance $d_{i,k}^{\mathrm{avg}}$ captures the mean viewing distance at which user i observes virtual content within the valid viewing distance during the kth planning window and is given by

$$d_{i,k}^{\mathrm{avg}} = \frac{1}{\lambda_{i,k}^{\mathrm{avg}} T} \sum_{t \in \mathcal{T}_k} d_{i,t} \cdot I\left(d_{i,t} \le d_{\max}\right). \quad (25)$$

2) XR Environmental Parameter: For environmental information, we mainly consider the density and overall rendering complexity of virtual objects located within each user's activity region. Let $(x_{i,t}, y_{i,t})$ denote the coordinates of user i at time slot t. The activity region of user i during planning window k is characterized by a bounding box, $\mathbf{B}_{i,k}$, defined as

$$\mathbf{B}_{i,k} = \left[\left(x_{i,k}^{\min}, y_{i,k}^{\min}\right), \left(x_{i,k}^{\max}, y_{i,k}^{\max}\right)\right] \quad (26)$$

where

$$x_{i,k}^{\min} = \min_{t \in \mathcal{T}_k} x_{i,t}, \quad x_{i,k}^{\max} = \max_{t \in \mathcal{T}_k} x_{i,t} \quad (27)$$

$$y_{i,k}^{\min} = \min_{t \in \mathcal{T}_k} y_{i,t}, \quad y_{i,k}^{\max} = \max_{t \in \mathcal{T}_k} y_{i,t}. \quad (28)$$

To predict the environmental information for the planning window k, we first estimate the user's future activity region by forecasting the bounding box based on historical observations. For user i, the predicted activity region during planning window k, denoted by $\hat{\mathbf{B}}_{i,k}$, is given by

$$\hat{\mathbf{B}}_{i,k} = \Lambda\left(\mathbf{B}_{i,k-1}, \mathbf{B}_{i,k-2}, \ldots, \mathbf{B}_{i,k-M}\right) \quad (29)$$

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

TABLE I
SIMULATION PARAMETERS

Parameter	Value	Parameter	Value
I	30	T	5000
τ_0	20 ms	$\hat{\tau}$	80 ms
ρ^{ren}	0.85 GFLOPs/bit	P_0	30 dBm
μ_1	1	μ_2	-0.5
μ_3	8	υ	0.03
ω_a	1	ω_b	0.5
$d_{\max}$	2.1 m	Δd	0.3 m
L	7	G	8
N	30	$Q_{\min}$	6.5
η_0	40	δ	1
ϵ_Q	0.01	M	2
$A_{\max}$	320 MHz	$B_{\max}$	857 GFLOPS.

where $\Lambda(\cdot)$ represents the prediction function and M denotes the number of historical planning windows used for prediction. The prediction can be performed using either linear regression models (e.g., the vector autoregressive model) or neural networks (e.g., the long short-term memory network) [48], [49].

Given predicted activity region $\hat{\mathbf{B}}_{i,k}$, the environmental parameter vector is defined as $\boldsymbol{f}_{i,k}^{\text{env}} = [\rho_{i,k}, c_{i,k}^{\text{avg}}]$, where $\rho_{i,k}$ denotes the virtual object density, defined as the number of virtual objects per square meter within region $\hat{\mathbf{B}}_{i,k}$, and $c_{i,k}^{\text{avg}}$ represents the average rendering complexity, measured in FLOPs required per virtual object within $\hat{\mathbf{B}}_{i,k}$. These parameters are given by

$$\rho_{i,k} = D\left(\hat{\mathbf{B}}_{i,k}\right), \quad c_{i,k}^{\text{avg}} = C\left(\hat{\mathbf{B}}_{i,k}\right) \tag{30}$$

where $D(\cdot)$ and $C(\cdot)$ represent the density and complexity extraction functions, respectively, which are determined by the spatial distribution and deployment characteristics of XR virtual objects.

3) DNN Training and Inference: Based on the extracted user viewing parameters, environmental parameters, collected user QoE, and resource consumption data, we train a DNN model, denoted by $F_{\text{DNN}}(\cdot)$, to assist with future resource demand estimation. Specifically, for user i in planning window k, the input vector to the DNN model consists of long-term user viewing parameters $\boldsymbol{f}_{i,k}^{\text{long}}$, environmental parameters $\boldsymbol{f}_{i,k}^{\text{env}}$, and QoE requirement $Q_{\min}$. The DNN inference process is then given by

$$\left[\hat{\alpha}_{i,k}, \hat{\beta}_{i,k}\right] = F_{\text{DNN}}\left(\boldsymbol{f}_{i,k}^{\text{long}}, \boldsymbol{f}_{i,k}^{\text{env}}, Q_{\min}\right) \tag{31}$$

where $\hat{\alpha}_{i,k}$ and $\hat{\beta}_{i,k}$ denote the predicted average bandwidth and computing resource consumption for user i during planning window k, respectively. The total estimated resource consumption is then given by $\hat{a}_k^{(0)} = \sum_{i\in\mathcal{I}} \hat{\alpha}_{i,k}$ and $\hat{b}_k^{(0)} = \sum_{i\in\mathcal{I}} \hat{\beta}_{i,k}$, which are used to initialize Algorithm 2.

V. PERFORMANCE EVALUATION

In this section, we conduct extensive simulations to evaluate the performance of the proposed mobility model and resource provisioning scheme using the real-world user trajectory datasets [50]. Simulations are conducted in MATLAB R2023b on a computer equipped with an Intel Core i7-12700H CPU (2.3 GHz) and 16-GB RAM, running 64-bit Windows 11. The main simulation parameters are given in Table I.

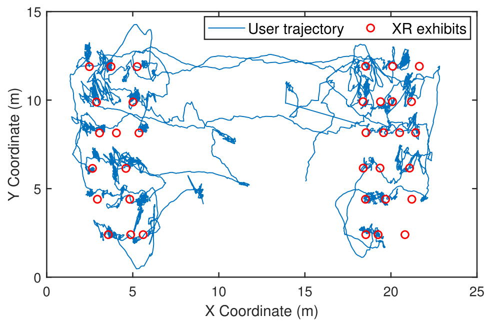


Fig. 4. Spatial deployment of XR exhibits and user trajectory for user ID 12.

TABLE II
SENSITIVITY WEIGHT $\zeta(d_{i,t})$

$d_{i,t}$(m)	(0, 0.3]	(0.3, 0.6]	(0.6, 0.9]	(0.9, 1.2]
ζ	1	0.85	0.7	0.55
$d_{i,t}$(m)	(1.2, 1.5]	(1.5, 1.8]	(1.8, 2.1]	(2.1, ∞)
ζ	0.4	0.15	0.05	0

A. Simulation Setup

The dataset provided in [50] contains the trajectories of ten museum visitors tracked via an ultrawideband localization system, along with the corresponding layout of the museum space. We use the "in vivo" trajectories, in which visitors moved freely within the study area. The trajectories span approximately 31 min and contain 22 340 time slots, each with a duration of approximately 84 ms. For evaluation, we divide the first 20 000 time slots into four consecutive planning windows, each consisting of 5000 time slots. The service area and virtual object layout are kept consistent with those in the dataset, where the service area is 25×15 m and contains 35 exhibits. Fig. 4 shows the spatial distribution of XR exhibits and depicts the trajectory of user ID 12.[5] To simulate a large-scale XR scenario, we augment the original dataset by generating 20 additional synthetic user trajectories. Specifically, for each original user, we extract a segment of their later trajectory and shift it to earlier time slots to simulate additional users.

One AP is placed at the center of the service area. At each time slot, the viewing distance is computed as the Euclidean distance between the user's coordinate and the center of the exhibit being viewed. A virtual object is rendered for a user only when the viewing distance is less than 2.1 m. Each physical exhibit is treated as a virtual object and is represented by a point cloud with seven available density versions. Rendering complexity $c(z_{i,t})$ for each point cloud is randomly sampled from a uniform distribution over $[0, 1]$. We set the rendering data size as $\psi(z_{i,t}, \gamma_{i,t}) = c(z_{i,t})\gamma_{i,t}\psi_0$, where $\psi_0 = 28.8$ MB denotes the base rendering data size. The transmission data size for rendered video frames is set as $\chi(\gamma_{i,t}) = \gamma_{i,t}\chi_0$, with $\chi_0 = 24$ MB representing the base transmission data size. For the QoE model, viewing distance sensitivity weight $\zeta(d_{i,t})$ used in visual quality evaluation is provided in Table II. Note that the values of $\zeta(d_{i,t})$ are determined following the same

[5]User ID 12 refers to the original identifier from the dataset, not sequential user numbering.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

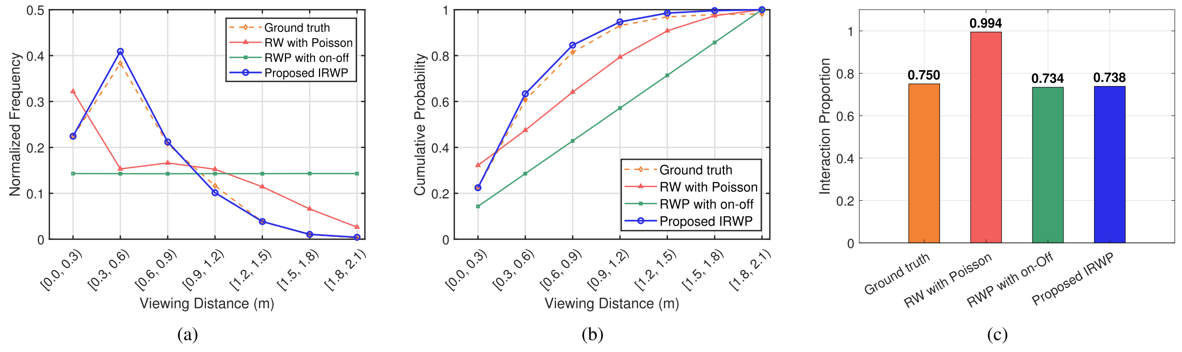


Fig. 5. User viewing behavior analysis when $k = 2$. (a) Normalized frequency distribution over viewing distance. (b) Cumulative probability distribution over viewing distance. (c) Average interaction proportion.

trend as in [15] but are adjusted accordingly to accommodate the different distance range in our dataset. In addition, the values of μ_1–μ_3 are empirically determined by balancing the contributions of visual quality, quality variation, and latency utility in our experiments.

Since the real-world dataset was collected in a relatively simple environment and over a short duration, we only use it to evaluate the performance of the proposed IRWP model and the SAA-based QoE estimation method. For evaluating the resource provisioning scheme, since the dataset is insufficient to train the DNN on certain input factors (e.g., virtual object density), we generate a simulated dataset with an expanded XR service environment, increased user numbers, and longer viewing trajectories. The details are provided in Section V-D.

B. Performance of IRWP Model

We first evaluate the accuracy of the proposed mobility model in characterizing user viewing behavior by analyzing viewing distance distribution and the average interaction proportion, and comparing these metrics with ground truth data. The following benchmark models are considered.

1) *RW With Poisson:* User mobility follows a random walk model, and rendering requests are generated based on a Poisson process.
2) *RWP With On–Off:* User mobility follows the RWP model, and rendering requests are modeled as an on–off process.
3) *Ground Truth:* The actual viewing trajectories for the next planning window are used.

The synthetic user trajectories for the benchmark models are generated as follows. First, historical viewing distance data from the previous planning window are discretized into integer levels (0–7) for the Poisson process or binary states (ON/OFF) for the on–off model. For example, distances exceeding 2.1 m are discretized to 0 (Poisson) or OFF (on–off). In the Poisson model, shorter distances correspond to higher integers, reflecting increased rendering intensity. Based on these discretized sequences, the Poisson arrival rates and ON/OFF state transition probabilities are derived using MATLAB's statistical fitting tools. These parameters are then used to generate future discrete rendering demands or states, which are subsequently mapped back to continuous viewing distances. For each time slot, if the user has a valid viewing distance, we randomly select a viewed object for that user.

TABLE III
PERFORMANCE COMPARISON IN TERMS OF KL DIVERGENCE AND MSE

Models	$k=2$	$k=3$	$k=4$
KL Divergence (nats)			
RW with Poisson	0.4827	1.0134	0.2641
RWP with on-off	0.7380	1.2356	0.4610
Proposed IRWP	**0.2431**	**0.8063**	**0.0292**
MSE			
RW with Poisson	0.010796	0.009851	0.008614
RWP with on-off	0.017064	0.014412	0.013462
Proposed IRWP	**0.000528**	**0.000925**	**0.000396**

Fig. 5 shows the user viewing behavior during planning window index 2. Specifically, Fig. 5(a) and (b) shows the normalized frequency and cumulative distribution over discretized viewing distances, respectively. Fig. 5(c) shows the average interaction proportion. From these figures, we observe that the proposed IRWP model closely aligns with the ground truth across all metrics, demonstrating its effectiveness in capturing user viewing behavior. In contrast, the Poisson model exhibits substantial discrepancies in both the viewing distance distribution and the interaction proportion. This is because the Poisson model reflects long-term statistical characteristics but fails to capture short-term dynamics, such as ON/OFF transitions and rapid fluctuations in viewing distance. Similarly, the RWP model shows significant inaccuracies in modeling the viewing distance, as it lacks the ability to reflect users' fine-grained viewing preferences. However, it achieves a comparable average interaction proportion to the IRWP model. This is because the RWP model's use of mobility pauses (e.g., waypoints), which partially mimics the ON/OFF interaction patterns observed in real-world user behavior.

We also evaluate the Kullback–Leibler (KL) divergence and mean squared error (MSE) of various models against the ground truth across different planning windows.[6] The results are presented in Table III, which shows that the proposed IRWP model not only achieves a better fit to the overall distribution but also maintains consistently low errors across all windows. Note that when $k = 3$, both the KL divergence and MSE increase sharply. This surge occurs because, during that planning window, users take a break and then return to

[6]User data from the first planning window are used for model parameter extraction. Therefore, the performance evaluation starts from the second planning window.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

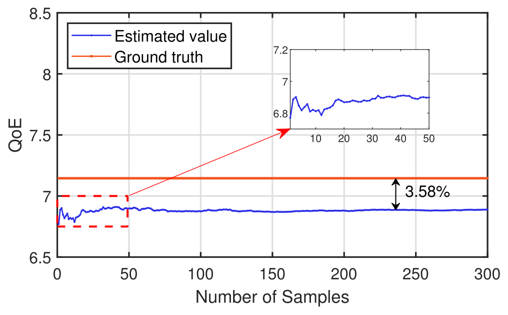


Fig. 6. Estimated QoE value versus number of samples when $a = 192$ MHz and $b = 103$ GFLOPS.

the museum, resulting in discontinuous behavior that adversely affects the model's characterization of user trajectories.

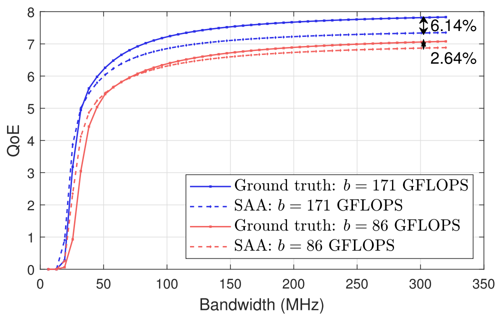
C. Performance of SAA-Based QoE Estimation

We next evaluate the effectiveness of the SAA-based QoE estimation method supported by the IRWP model. Fig. 6 illustrates the estimated QoE value versus the number of samples. From the figure, we have the following observations. First, the estimated QoE becomes stable as the number of samples increases. Second, the estimation closely approximates the ground truth, with an average deviation of approximately 3.58%, which validates the accuracy of the proposed mobility model and the effectiveness of the estimation method. Third, even with a sufficiently large number of samples, a small gap remains between the estimated and ground truth QoE values. This error is due to the inherent variability in user mobility and interaction behaviors across different time windows, which cannot be fully captured through trajectory sampling.

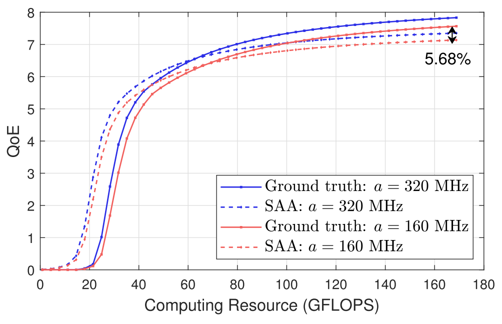
We further analyze the impact of resource capacity on the accuracy of the SAA-based QoE estimation, with results shown in Fig. 7(a) and (b). As shown in both figures, as bandwidth and computing resources are sufficiently provisioned, the SAA-based method tends to underestimate user QoE, with accuracy gaps of approximately 6.14% and 5.68%, respectively. This underestimation is probably due to the IRWP model slightly overestimating the rendering demand in certain time slots, which in turn leads to a conservative QoE estimate given the resource provisioning value. This hypothesis is supported by the observations in Fig. 5(a), where the IRWP model slightly overestimates the probability of users experiencing shorter viewing distances compared to the ground truth.

D. Performance of Resource Provisioning

In this section, we evaluate the performance of the proposed resource provisioning scheme. As previously mentioned, the dataset released in [50] is not sufficient for this part of the analysis due to its limited scale and diversity. Therefore, we generate a synthetic dataset for simulation. To be specific, we simulate an XR service area of 200×200 m, in which $Z = 200$ virtual objects are nonuniformly distributed. Trajectories are generated for $I = 100$ XR users over $K = 10$ planning windows. To create the training dataset for the DNN model, we vary system parameters such as resource capacity, rendering complexity, and object distributions, followed by



(a)



(b)

Fig. 7. QoE performance versus resource capacity. (a) Bandwidth. (b) Computing resource.

generating the corresponding QoE values. A total of 95 100 data samples are collected. For model training, we use the neural network fitting function (fitnet) in MATLAB, which is designed for regression tasks. The network consists of a two-layer feedforward structure with two hidden neurons using the tansig activation, followed by a linear output layer. The model is trained using the Levenberg–Marquardt algorithm, where the MSE is used as the performance function. The maximum epoch is set to 1000, and the training terminates once the gradient falls below 10^{-7}. The dataset is randomly split into 70% for training, 15% for validation, and 15% for testing. For environmental information prediction, we adopt a vector autoregression model of order 2 to do estimation. A sliding window of 250 slots is used, and the prediction is iteratively performed until the information for all $T = 5000$ future slots is estimated.

We first evaluate the convergence performance of the proposed coordinate descent search algorithm, which utilizes a pretrained DNN to provide an initial search point. For benchmark, we consider the randomized initialization (RI) approach, where the initial values of $a^{(0)}$ and $b^{(0)}$ are randomly selected from the feasible sets $[0, A_{\max}]$ and $[0, B_{\max}]$, respectively. Fig. 8 compares the convergence trajectories of computing resource consumption and weighted resource cost with respect to the number of iterations. Key observations are as follows. The proposed algorithm (blue curves) converges in substantially fewer iterations than RI. This acceleration is due to the DNN's ability to provide high-quality initial points that are close to the optimal solution, thereby reducing the search space. In contrast, the RI method exhibits slower convergence due to its uninformed initialization, which results in unnecessary exploration of suboptimal regions.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

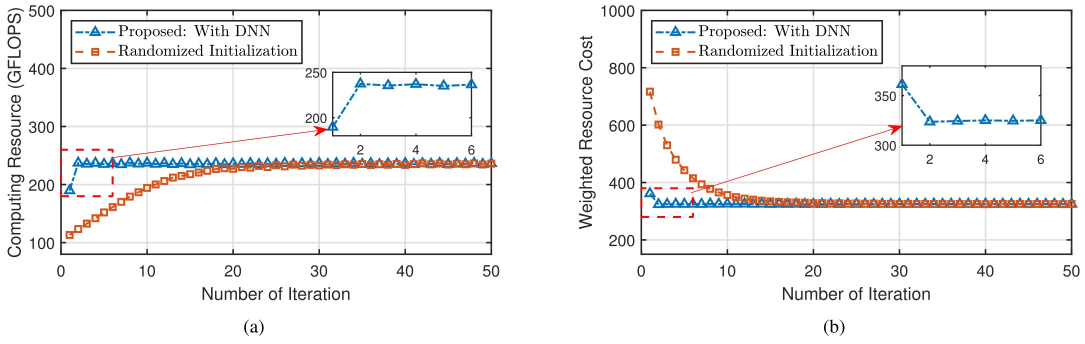


Fig. 8. Convergence of the algorithm. (a) Computing resource provisioning. (b) Weighted resource cost.

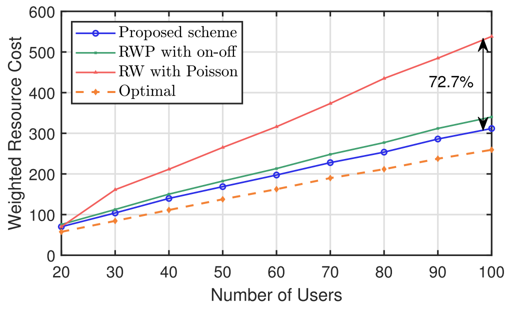


Fig. 9. Weighted resource cost versus number of users.

We then evaluate the effectiveness of the proposed resource provisioning scheme. We consider benchmark schemes that use different traffic or mobility models while sharing the same resource provisioning algorithm as ours. Fig. 9 shows the weighted resource cost under different numbers of users. The result indicates that the proposed scheme consistently achieves performance closest to that of the optimal scheme. Specifically, when the number of users reaches 100, the proposed scheme reduces the weighted resource cost by up to 72.7% and 9.1% compared to the schemes based on *RW with Poisson* and *RWP with on–off* models, respectively. This performance gain is attributed to the proposed IRWP model, which effectively captures key mobility and interaction characteristics of XR users. As a result, the generated viewing trajectories are more representative in estimating users' spatiotemporal service demands, enabling more accurate QoE estimation and more efficient resource provisioning.

Finally, we evaluate the actual runtime of the proposed resource provisioning algorithm to demonstrate its feasibility in practice. Fig. 10(a)–(c) shows the actual runtime of Algorithm 2 with respect to the number of samples considered in the QoE estimation, the computing resource capacity, and the convergence tolerance, respectively. The 95% confidence interval error bars, based on 100 experimental runs, are included. We observe that the actual runtime is feasible in practice when compared to the planning window timescale. Moreover, the runtime increases approximately linearly with the number of samples, as well as with the logarithm of $B_{\max}$ and $1/\epsilon_Q$, which aligns well with the complexity analysis discussed in Section IV-D3. We also evaluate the actual runtime of the proportional-sharing scheduling policy used in this work. As shown in Fig. 10(d), the actual runtime increases almost linearly with the number of users, which is consistent with the algorithmic complexity analysis.

VI. Conclusion

In this article, we have proposed an efficient resource provisioning scheme for edge-assisted XR services. Specifically, we have presented a mobility model tailored for XR users that incorporates their interactive viewing behaviors to enable accurate characterization of dynamic service demands. Then, we have designed a user trajectory sampling-based algorithm that efficiently determines the minimum required network resources for user QoE satisfaction. The proposed scheme allows flexible network resource provisioning for XR services, thereby reducing long-term operational costs. For future work, we will develop an analytical framework to study the impact of mobility model parameters on both effective network capacity and user QoE.

Appendix A
Proportional-Sharing Scheduling Policy

The proportional-sharing scheduling policy allocates both radio bandwidth and computing resources to users in proportion to their instantaneous transmission and rendering workloads. Additionally, the rendering quality assigned to each user depends only on the user's current viewing distance. Specifically, for user i at slot t, the quality level is determined by

$$\gamma_{i,t} = \begin{cases} \max\left\{0,\ L - \left\lfloor \dfrac{d_{i,t} L}{\Delta d} \right\rfloor\right\}, & \text{if } d_{i,t} \le d_{\max} \\ 0, & \text{otherwise} \end{cases} \tag{32}$$

where L is the highest possible quality level. This rule ensures that users with larger viewing distances are assigned lower rendering quality levels.

The bandwidth allocated to user i at slot t is given by

$$\alpha_{i,t} = a \times \frac{w_{i,t}^{\text{tx}}}{\sum_{j=1}^{I} w_{j,t}^{\text{tx}}} \tag{33}$$

where $w_{i,t}^{\text{tx}} = \chi(\gamma_{i,t}) / \log_2(1 + P_0 G_i h_{i,t}/\sigma^2)$ denotes the transmission weight, which reflects the effective data rate of each user. Similarly, the computing resource allocated to user i at slot t is given by

$$\beta_{i,t} = b \times \frac{\psi\left(z_{i,t}, \gamma_{i,t}\right)}{\sum_{j=1}^{I} \psi\left(z_{j,t}, \gamma_{j,t}\right)}. \tag{34}$$

This resource allocation policy ensures that, at each time slot, the rendering and transmission latency experienced by all users is approximately equal.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

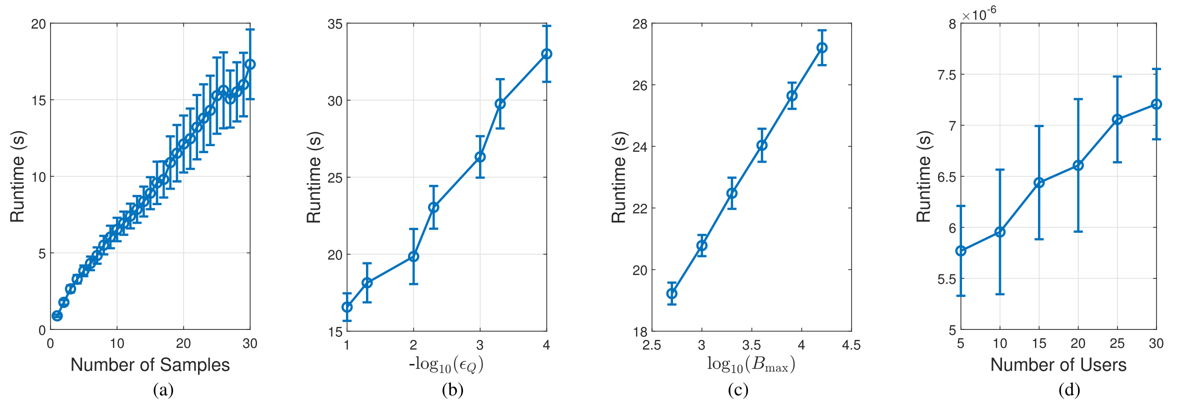


Fig. 10. Runtime analysis under different system configurations. (a) Varied number of samples. (b) Varied convergence tolerance. (c) Varied computing resource capacity. (d) Per-slot scheduling runtime.

Appendix B
Convexity Proof

Proving the convexity of (14b) with respect to both a and b is equivalent to proving the concavity of $Q_{i,t}(a,b)$. Since both $V_{i,t}$ and $\Delta V_{i,t}$ depend only on rendering quality $\gamma_{i,t}$, which in turn is determined by the user's viewing distance, they are independent of a and b. Therefore, we focus on analyzing the concavity of $U(\tau^{\text{rt}}_{i,t}(a,b))$. We begin by computing the first-order partial derivative of $U(\tau^{\text{rt}}_{i,t}(a,b))$ with respect to a, given by

$$\begin{aligned}\frac{\partial U\left(\tau^{\text{rt}}_{i,t}(a,b)\right)}{\partial a} &= -\frac{\upsilon\exp\left(\upsilon\left(\tau^{\text{rt}}_{i,t}-\hat{\tau}\right)\right)}{\left[1+\exp\left(\upsilon\left(\tau^{\text{rt}}_{i,t}-\hat{\tau}\right)\right)\right]^2}\times\frac{\partial\tau^{\text{rt}}_{i,t}(a,b)}{\partial a}\\ &= -\frac{\upsilon\exp\left(\upsilon\left(\tau^{\text{rt}}_{i,t}-\hat{\tau}\right)\right)}{\left[1+\exp\left(\upsilon\left(\tau^{\text{rt}}_{i,t}-\hat{\tau}\right)\right)\right]^2}\times\frac{\partial\tau^{\text{tx}}_{i,t}(a)}{\partial a}\end{aligned}\qquad(35)$$

where (35) follows by excluding terms that are independent of a. Given the proportional-sharing scheduling policy described in (33) and (34), the transmission latency can be given by

$$\tau^{\text{tx}}_{i,t}(a) = \frac{1}{a}\sum_{j=1}^{I} w^{\text{tx}}_{j,t}\quad \forall i\in\mathcal{I}.\qquad(36)$$

Thus, the first-order derivative of $\tau^{\text{tx}}_{i,t}(a)$ with respect to a is given by

$$\frac{\partial\tau^{\text{tx}}_{i,t}(a)}{\partial a} = -\frac{1}{a^2}\sum_{j=1}^{I} w^{\text{tx}}_{j,t}\quad \forall i\in\mathcal{I}.\qquad(37)$$

Next, we derive the second-order partial derivative of $U(\tau^{\text{rt}}_{i,t}(a,b))$ with respect to a. Since b is independent of a, we treat it as constant during differentiation. For convenience, we define $f(a) \triangleq \exp[\upsilon(\tau^{\text{rt}}_{i,t}-\hat{\tau})]$. The second-order partial derivative, denoted by $\partial^2 U/\partial a^2$, is then given by

$$\begin{aligned}\frac{\partial^2 U}{\partial a^2} = &\left(\frac{\partial f(a)}{\partial a}\times\frac{1}{a^2}\left[\frac{1}{(1+f(a))^2}-\frac{2f(a)}{(1+f(a))^3}\right]\right.\\ &\left.-\frac{2f(a)}{(1+f(a))^2a^3}\right)\times\upsilon\sum_{j=1}^{I} w^{\text{tx}}_{j,t}.\end{aligned}\qquad(38)$$

The first-order derivative of $f(a)$ is given by

$$\frac{\partial f(a)}{\partial a} = -\upsilon\sum_{j=1}^{I} w^{\text{tx}}_{j,t}\times\frac{f(a)}{a^2}.\qquad(39)$$

Substituting (39) into (38), we obtain

$$\begin{aligned}\frac{\partial^2 U}{\partial a^2} = &-\upsilon\sum_{j=1}^{I} w^{\text{tx}}_{j,t}\times\frac{f(a)}{(1+f(a))^2a^3}\\ &\times\left(\underbrace{\upsilon\sum_{j=1}^{I} w^{\text{tx}}_{j,t}a^{-1}\left[1-\frac{2f(a)}{1+f(a)}\right]}+2\right).\end{aligned}\qquad(40)$$

To determine the sign of the second-order derivative in (40), we only need to examine the expression within the parentheses. Substituting (36) into the under-braced term in (40), we obtain

$$\begin{aligned}&\upsilon\tau^{\text{tx}}_{i,t}(a)\left(1-\frac{2}{1+f(a)^{-1}}\right)\\ &=\upsilon\tau^{\text{tx}}_{i,t}(a)\left(1-\frac{2}{1+\exp\left[-\upsilon\left(\tau^{\text{rt}}_{i,t}(a,b)-\hat{\tau}\right)\right]}\right)\\ &=\upsilon\tau^{\text{tx}}_{i,t}(a)\left(1-\frac{2}{1+\exp\left[-\upsilon\left(\tau^{\text{tx}}_{i,t}(a)+\tau_0+\tau^{\text{ren}}_{i,t}(b)-\hat{\tau}\right)\right]}\right).\end{aligned}\qquad(41)$$

The expression in (41) is a nonincreasing function of $\tau^{\text{tx}}_{i,t}(a)$. Specifically, as $\tau^{\text{tx}}_{i,t}(a)\to 0$, the value inside the parentheses in (41) tends to zero. Conversely, as $\tau^{\text{tx}}_{i,t}(a)\to+\infty$, it approaches positive infinity. Therefore, the quantity inside the parentheses in (40) is always greater than 2, regardless of the value of $\tau^{\text{tx}}_{i,t}(a)$. As a result, the second-order derivative satisfies $\partial^2 U/\partial a^2 < 0$ for all a. This completes the proof.

References

[1] X. Shen et al., "Toward immersive communications in 6G," *Frontiers Comput. Sci.*, vol. 4, pp. 1–45, Jan. 2023.

[2] Extended Reality (XR) in 5G; Version 18.0.0, Standard TR 26.928, 3GPP, Sophia Antipolis, France, Mar. 2023.

[3] Support of 5G Glass-Type Augmented Reality/Mixed Reality (AR/MR) Devices; Version 18.1.0, Standard TR 26.998, 3GPP, Sophia Antipolis, France, Mar. 2024.

[4] J. v. d. Hooft et al., "A tutorial on immersive video delivery: From omnidirectional video to holography," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 2, pp. 1336–1375, 2nd Quart., 2023.

[5] Y. Guan, X. Hou, N. Wu, B. Han, and T. Han, "MetaStream: Live volumetric content capture, creation, delivery, and rendering in real time," in *Proc. 29th Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2023, pp. 1–15.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

[6] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proc. IEEE*, vol. 107, no. 8, pp. 1738–1762, Aug. 2019.

[7] Y. Pei et al., "Interaction-aware resource reservation for edge-assisted mobile extended reality," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Aug. 2025, pp. 1–6.

[8] S. Wijethilaka and M. Liyanage, "Survey on network slicing for Internet of Things realization in 5G networks," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 2, pp. 957–994, 2nd Quart., 2021.

[9] X. Shen et al., "AI-assisted network-slicing based next-generation wireless networks," *IEEE Open J. Veh. Technol.*, vol. 1, pp. 45–66, 2020.

[10] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2429–2453, 3rd Quart., 2018.

[11] M. Zangooei, M. Golkarifard, M. Rouili, N. Saha, and R. Boutaba, "Flexible RAN slicing in open RAN with constrained multi-agent reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 2, pp. 280–294, Feb. 2024.

[12] S. Zhang, Z. Qian, Z. Luo, J. Wu, and S. Lu, "Burstiness-aware resource reservation for server consolidation in computing clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 4, pp. 964–977, Apr. 2016.

[13] C. Bettstetter, H. Hartenstein, and X. Pérez-Costa, "Stochastic properties of the random waypoint mobility model," *Wireless Netw.*, vol. 10, no. 5, pp. 555–567, Sep. 2004.

[14] W. L. Tan, W. C. Lau, O. Yue, and T. H. Hui, "Analytical models and performance evaluation of drive-thru internet systems," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 1, pp. 207–222, Jan. 2011.

[15] A. Zhang, C. Wang, B. Han, and F. Qian, "YuZu: Neural-enhanced volumetric video streaming," in *Proc. 19th USENIX Symp. Networked Syst. Design Implement.*, 2022, pp. 137–154.

[16] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "Holistic network virtualization and pervasive network intelligence for 6G," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 1–30, 1st Quart., 2022.

[17] F. Chiariotti, M. Drago, P. Testolina, M. Lecci, A. Zanella, and M. Zorzi, "Temporal characterization and prediction of VR traffic: A network slicing use case," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 3890–3908, May 2024.

[18] S. Mondal and E. Wong, "User head movement-predictive XR in immersive H2M collaborations over future enterprise networks," *IEEE Internet Things J.*, vol. 12, no. 18, pp. 38901–38913, Sep. 2025.

[19] L. Mastrandrea, A. Priviero, G. Scarano, S. Colonnese, and T. Cattai, "Simulating extended reality traffic: An empirical model from user behavior to network packets," *Comput. Commun.*, vol. 241, Sep. 2025, Art. no. 108244.

[20] Y. Chen, S. Omoma, H. Kwon, H. Inaltekin, and M. Gorlatova, "Quantifying and exploiting VR frame correlations: An application of a statistical model for viewport pose," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 11466–11482, Dec. 2024.

[21] Q. Ye, W. Shi, K. Qu, H. He, W. Zhuang, and X. Shen, "Joint RAN slicing and computation offloading for autonomous vehicular networks: A learning-assisted hierarchical approach," *IEEE Open J. Veh. Technol.*, vol. 2, pp. 272–288, 2021.

[22] H. Zhang and V. W. S. Wong, "A two-timescale approach for network slicing in C-RAN," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 6656–6669, Jun. 2020.

[23] Z. Ming, H. Yu, and T. Taleb, "Federated deep reinforcement learning for prediction-based network slice mobility in 6G mobile networks," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 11937–11953, Dec. 2024.

[24] Y. Hua, R. Li, Z. Zhao, X. Chen, and H. Zhang, "GAN-powered deep distributional reinforcement learning for resource management in network slicing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 334–349, Feb. 2020.

[25] J. Zhang, S. Chen, X. Wang, and Y. Zhu, "DeepReserve: Dynamic edge server reservation for connected vehicles with deep reinforcement learning," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2021, pp. 1–10.

[26] Y. Cai, P. Cheng, Z. Chen, M. Ding, B. Vucetic, and Y. Li, "Deep reinforcement learning for online resource allocation in network slicing," *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 7099–7116, Jun. 2024.

[27] G. Chen, S. Qi, F. Shen, Q. Zeng, and Y.-D. Zhang, "Information-aware driven dynamic LEO-RAN slicing algorithm joint with communication, computing, and caching," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 5, pp. 1044–1062, May 2024.

[28] V. Sciancalepore, X. Costa-Perez, and A. Banchs, "RL-NSB: Reinforcement learning-based 5G network slice broker," *IEEE/ACM Trans. Netw.*, vol. 27, no. 4, pp. 1543–1557, Aug. 2019.

[29] H. Nan, R. Li, X. Zhu, J. Ma, and K. Xue, "Spatio-temporal identity multi-graph convolutional network for traffic prediction in the metaverse," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 3, pp. 669–679, Mar. 2024.

[30] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?," *IEEE Trans. Commun.*, vol. 67, no. 10, pp. 7331–7376, Oct. 2019.

[31] M. Li, J. Gao, C. Zhou, L. Zhao, and X. Shen, "Digital-twin-empowered resource allocation for on-demand collaborative sensing," *IEEE Internet Things J.*, vol. 11, no. 23, pp. 37942–37958, Dec. 2024.

[32] C. Lin, C. Kong, and S. Lucey, "Learning efficient point cloud generation for dense 3D object reconstruction," in *Proc. 32nd AAAI Conf. Artif. Intell.*, New Orleans, LA, USA, 2018, pp. 7114–7121.

[33] K. Lee, J. Yi, Y. Lee, S. Choi, and Y. M. Kim, "GROOT: A real-time streaming system of high-fidelity volumetric videos," in *Proc. Int. Conf. Mobile Comput. Netw.*, 2020, pp. 1–14.

[34] R. Xie, J. Fang, J. Yao, X. Jia, and K. Wu, "Sharing-aware task offloading of remote rendering for interactive applications in mobile edge computing," *IEEE Trans. Cloud Comput.*, vol. 11, no. 1, pp. 997–1010, Jan. 2023.

[35] Y. Pei, M. Li, X. Huang, and X. Shen, "QoE-aware volumetric video caching and rendering for mobile extended reality services," *IEEE Internet Things J.*, vol. 12, no. 12, pp. 21852–21865, Jun. 2025.

[36] D. Yu, R. Chen, X. Li, M. Xiao, G. Zhang, and Y. Liu, "A GPU-enabled real-time framework for compressing and rendering volumetric videos," *IEEE Trans. Comput.*, vol. 73, no. 3, pp. 789–800, Mar. 2024.

[37] O. Simeone, Y. Bar-Ness, and U. Spagnolini, "Pilot-based channel estimation for OFDM systems by tracking the delay-subspace," *IEEE Trans. Wireless Commun.*, vol. 3, no. 1, pp. 315–325, Jan. 2004.

[38] P. Cui, S. Han, L. Li, B. Zhou, X. Xu, and P. Zhang, "Roundtrip interaction delay analysis of immersive communications: A stochastic network calculus perspective," *IEEE Trans. Wireless Commun.*, vol. 24, no. 3, pp. 2188–2202, Mar. 2025.

[39] H. Cao, J. Xu, D. Li, L. Shangguan, Y. Liu, and Z. Yang, "Edge assisted mobile semantic visual SLAM," *IEEE Trans. Mobile Comput.*, vol. 22, no. 12, pp. 6985–6999, Dec. 2023.

[40] Y. Zhu, Y. Huang, X. Qiao, J. Tang, and X. Su, "HiVAT: Improving QoE for hybrid video streaming service with adaptive transcoding," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 11209–11226, Dec. 2024.

[41] X. Zhang, S. Sen, D. Kurniawan, H. Gunawi, and J. Jiang, "E2E: Embracing user heterogeneity to improve quality of experience on the web," in *Proc. ACM Special Interest Group Data Commun.*, Aug. 2019, pp. 289–302.

[42] J. Wu, X. Zhuang, M. Tang, and L. Gao, "QoE-aware offloading and resource allocation for MEC-empowered AIGC services," *IEEE Trans. Mobile Comput.*, vol. 24, no. 10, pp. 9664–9682, Oct. 2025.

[43] C. Bettstetter, G. Resta, and P. Santi, "The node distribution of the random waypoint mobility model for wireless ad hoc networks," *IEEE Trans. Mobile Comput.*, vol. 2, no. 3, pp. 257–269, Jul. 2003.

[44] A. J. Kleywegt, A. Shapiro, and T. Homem-De-Mello, "The sample average approximation method for stochastic discrete optimization," *SIAM J. Optim.*, vol. 12, no. 2, pp. 479–502, Jan. 2002.

[45] D. P. Bertsekas, *Nonlinear Programming*. Cambridge, MA, USA: MIT Press, 1999.

[46] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2009.

[47] L. Huang, S. Bi, and Y.-J.-A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2581–2593, Nov. 2020.

[48] J. Wang, X. Su, Y. Huang, H. Lai, W. Qian, and S. Zhang, "CLinear: An interpretable deep time-series forecasting model for periodic time series," *IEEE Internet Things J.*, vol. 12, no. 11, pp. 17364–17376, Jun. 2025.

[49] J. Wang et al., "Generative AI enabled robust data augmentation for wireless sensing in ISAC networks," *IEEE J. Sel. Areas Commun.*, early access, Sep. 24, 2025, doi: 10.1109/JSAC.2025.3613672.

[50] F. Hachem, D. Vecchia, M. L. Damiani, and G. P. Picco, "UWB trajectories and fine-grained stop-move detection: A museum dataset," Tech. Rep., 2025, doi: 10.5281/zenodo.14918763.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.

Yingying Pei (Graduate Student Member, IEEE) received the B.E. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2014, and the M.Sc. degree from the University of Macau, Macau, China, in 2019. She is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada.

Her research interests include network resource optimization, mobile edge computing, and immersive communications.

Mingcheng He (Member, IEEE) received the B.Sc. and M.Eng. degrees from Shanghai Jiao Tong University, Shanghai, China, in 2017 and 2020, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2024.

He is currently a Post-Doctoral Fellow with the University of Waterloo. His research interests include network slicing in satellite–terrestrial integration networks and artificial intelligence for future wireless networks.

Shisheng Hu (Member, IEEE) received the B.Eng. and M.A.Sc. degrees from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2018 and 2021, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2025.

He is currently a Post-Doctoral Fellow with the University of Waterloo. His research interests include edge intelligence, integrated sensing and communications, and machine learning for wireless networks.

Xinyu Huang (Member, IEEE) received the B.E. degree in Qian Xuesen Experimental Class from Xidian University, Xi'an, China, in 2018, the M.S. degree in information and communications engineering from Xi'an Jiaotong University, Xi'an, in 2021, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2025.

He is currently a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of Waterloo. His research interests include digital agents, multimedia communications, integrated sensing and communications, and network management.

Conghao Zhou (Member, IEEE) received the B.Eng. degree from Northeastern University, Shenyang, China, in 2017, the M.Sc. degree from the University of Illinois at Chicago, Chicago, IL, USA, in 2018, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2022.

He was a Post-Doctoral Fellow with the University of Waterloo. He is currently a Professor with the School of Telecommunications Engineering, Xidian University, Xi'an, China. His current research interests include immersive communications, AI for networking, and space–air–ground integrated networks.

Weihua Zhuang (Fellow, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from Dalian Marine University, Dalian, China, in 1982 and 1985, respectively, and the Ph.D. degree in electrical engineering from the University of New Brunswick, Fredericton, NB, Canada, in 1993.

She is a University Professor and a University Research Chair in electrical and computer engineering at the University of Waterloo, Waterloo, ON, Canada. Her current research focuses on network architecture, algorithms and protocols, and service provisioning in future telecommunication systems.

Dr. Zhuang is a fellow of the Royal Society of Canada (RSC), Canadian Academy of Engineering (CAE), and the Engineering Institute of Canada (EIC). She is an elected member of the Board of Governors of the IEEE Vehicular Technology Society (VTS). She was a recipient of the Women's Distinguished Career Award in 2021 from the IEEE Vehicular Technology Society, the R.A. Fessenden Award in 2021 from IEEE Canada, the Award of Merit in 2021 from the Federation of Chinese Canadian Professionals (Ontario), and the Technical Recognition Award in Ad Hoc and Sensor Networks in 2017 from the IEEE Communications Society. She was a Tier I Canada Research Chair in wireless communication network from 2010 to 2024, the VTS President from 2023 to 2024, the Editor-in-Chief of IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from 2007 to 2013, an Editor of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS from 2005 to 2009, the General Co-Chair of the IEEE/CIC International Conference on Communications in China (ICCC) 2021, the Technical Program Committee (TPC) Chair/Co-Chair of the IEEE Vehicular Technology Conference 2017 Fall and 2016 Fall, the TPC Symposia Chair of the IEEE Globecom 2011, and an IEEE Communications Society Distinguished Lecturer from 2008 to 2011.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990.

He is a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research focuses on network resource management, wireless network security, the Internet of Things, AI for networks, and vehicular networks.

Dr. Shen is a registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Member, an International Fellow of the Engineering Academy of Japan, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society. He received the "West Lake Friendship Award" from Zhejiang Province in 2023, the President's Excellence in Research from the University of Waterloo in 2022, Canadian Award for Telecommunications Research from Canadian Society of Information Theory (CSIT) in 2021, the R.A. Fessenden Award in 2019 from IEEE Canada, the Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) in 2019, the James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, the Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society (ComSoc), and the Technical Recognition Award from Wireless Communications Technical Committee (2019) and AHSN Technical Committee (2013). He has also received the Excellent Graduate Supervision Award in 2006 from the University of Waterloo and the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He is the Past President of the IEEE Communications Society. He was the Vice President for Technical and Educational Activities, the Vice President for Publications, the Member-at-Large on the Board of Governors, the Chair of the Distinguished Lecturer Selection Committee, and a member of the IEEE Fellow Selection Committee of the ComSoc. He served as the Editor-in-Chief for IEEE INTERNET OF THINGS JOURNAL, *IEEE Network*, and *IET Communications*.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 18,2026 at 00:15:45 UTC from IEEE Xplore. Restrictions apply.