Towards an AI-Enabled Metaverse: Architecture, Resource Allocation, and Multimodal Benchmarking

Zijian Long

Thesis submitted to the University of Ottawa
in partial Fulfillment of the requirements for the
Doctorate in Philosophy degree in Electrical Engineering and Computer Science

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

The metaverse, envisioned as a paradigm for next-generation digital environments, remains fundamentally constrained by isolated, pre-defined, and reactive virtual systems. Artificial intelligence (AI) provides a conceptual and technical foundation for overcoming these limitations by enabling persistent perception, autonomous decision-making, and adaptive coordination. This thesis investigates how AI can be systematically embedded into metaverse systems from a system-level perspective. It begins with a comprehensive analysis of metaverse network traffic characteristics, examining the necessity, performance implications, and trade-offs between remote rendering and local rendering. Building on these empirical insights, the thesis proposes a unified thing–edge–cloud architecture that coordinates heterogeneous intelligent agents operating across network layers.

Within this architecture, the thesis addresses two core challenges in AI-enabled metaverse systems: adaptive resource allocation and cognitive scene understanding. At the edge layer, adaptive bandwidth allocation for immersive streaming is formulated as a cooperative multi-agent decision-making problem under dynamic network conditions. A Multi-Agent Soft Actor Critic (MASAC)–based strategy is developed and yields consistent improvements in user Quality of Experience (QoE) of at least 14% compared with representative streaming baselines. At the cloud layer, the thesis introduces a metaverse-oriented benchmarking framework for evaluating the scene understanding capabilities of Multimodal Large Language Models (MLLMs). Thirteen representative MLLMs are assessed through a pairwise comparison protocol under an MLLM-as-a-judge paradigm guided by metaverse-specific semantic criteria. Human expert validation confirms the robustness of the proposed benchmark, achieving an agreement rate of 87.6% with human consensus. Overall, this thesis establishes an integrated methodological and architectural foundation for the design, optimization, and evaluation of AI-enabled metaverse systems.

Acknowledgements

Foremost, I would like to express my deepest and most sincere gratitude to my supervisor, Professor Abdulmotaleb El Saddik, for his unwavering support, patient guidance, and insightful mentorship throughout the entire course of my doctoral study. His academic rigor, visionary thinking, and commitment to excellence have not only shaped the direction of my research but also profoundly influenced the way I approach problems, think critically, and persevere in the face of challenges. Beyond academia, his encouragement and understanding during difficult moments in my personal life have meant a great deal to me. I truly could not have completed this journey without his steadfast support and inspiring leadership.

I am also especially grateful to Dr. Haiwei Dong, whose enlightening advice, constructive feedback, and technical insights have been invaluable to my research development. His clarity of thought and willingness to always make time for discussion, regardless of how busy he is, have greatly enriched my understanding of the field. I owe many breakthroughs and critical reflections in my work to his generous guidance.

I would like to extend my heartfelt appreciation to all my fellow lab members in the Multimedia Communications Research Laboratory (MCRLab). Their intellectual companionship, collaborative spirit, and camaraderie have created a vibrant and supportive research environment. Whether it was through spirited discussions in the lab, late-night debugging sessions, or casual chats that lifted my spirits, each of them contributed in their own unique way to this memorable academic journey. I am deeply proud and honored to have been part of the MCRLab family. The shared memories, laughter, and collective achievements will remain with me forever.

My sincere thanks also go to my friends, both near and far, for their emotional support, encouragement, and friendship during this demanding period. Their constant belief in me served as a source of strength and resilience, especially during the times when I doubted myself.

Finally, and most importantly, I wish to express my profound and everlasting

gratitude to my family, including my parents, my grandparents, and my girlfriend, for their unconditional love, patience, and sacrifices throughout the years. Their faith in me has been my foundation. Their financial support, emotional encouragement, and moral guidance have sustained me during the most difficult phases of my doctoral journey. No words are enough to fully convey my love and appreciation for them. This achievement is as much theirs as it is mine.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

6G Sixth Generation Mobile Network

ABR Adaptive BitRate

ACF Autocorrelation Function

ACU adaptive CPU usage

ADF Augmented Dickey–Fuller

AI Artificial Intelligence

AR Augmented Reality

ARMA Autoregressive Moving Average

BBR Bottleneck Bandwidth and Round-trip Time

CTDE Centralized Training with Decentralized Execution

DAOs Decentralized Autonomous Organizations

DASH Dynamic Adaptive Streaming over HTTP

Dec-POMDP Decentralized Partially Observable Markov Decision Process

DL Deep Learning

DQN Deep Q-Network

DRL Deep Reinforcement Learning

FedAvg Federated Averaging

FL Federated Learning

FOV Field of View

GAN Generative Adversarial Network

GCC Google Congestion Control

GCNs Graph Convolutional Networks

HLS HTTP Live Streaming

HPU Holographic Processing Unit

HTTP Hypertext Transfer Protocol

ISAC Independent Soft Actor-Critic

LDP Local Differential Privacy

LRD Long-Range Dependence

MADDPG Multi-Agent Deep Deterministic Policy Gradient

MADRL Multi-Agent Deep Reinforcement Learning

MASAC Multi-Agent Soft Actor-Critic

MFRL Mean Field Reinforcement Learning

MOS Mean Opinion Score

MR Mixed Reality

NFT Non-Fungible Token

PACF Partial Autocorrelation Function

PDF Probability Density Function

QoE Quality of Experience

Q–Q Plot Quantile–Quantile Plot

RTP Real-Time Transport Protocol

SDN Software-Defined Networking

SRD Short-Range Dependence

UE Unreal Engine

UPS Unreal Pixel Streaming

VR Virtual Reality

XR Extended Reality

Chapter 1

Introduction

1.1 Introduction

The concept of the metaverse first entered public consciousness through speculative fiction, most notably in Neal Stephenson's 1992 novel Snow Crash, which depicted a virtual environment coexisting alongside the physical world [1]. Although originally a fictional construct, this vision captured widespread imagination and gradually transitioned into an object of academic inquiry and public discourse. Early experiments with online virtual environments, such as Second Life, represented initial attempts to materialize elements of this imagined world and provided early insights into digital embodiment and social interaction in virtual spaces [2]. Over time, the metaverse has evolved from a narrative metaphor into a conceptual framework for understanding how human presence, identity, and interaction are reconfigured in increasingly digitized environments.

Today, the metaverse is commonly understood as a multidimensional socio-digital construct encompassing immersive, persistent, and interconnected virtual environments. Rather than referring to a single platform or proprietary technology, it denotes a broader digital ecosystem that spans multiple domains, including entertainment, education, commerce, work, and social life [3]. Often regarded as a potential next stage in the evolution of the internet, the metaverse signifies a shift in user engagement from passive information consumption toward active participation in shared virtual

worlds [4]. This transformation reflects deeper changes in how individuals inhabit digital spaces, enabling new forms of social interaction, collective creation, and community formation that blur the boundaries between the virtual and the physical.

As the metaverse gains cultural and economic prominence, it has attracted increasing attention from both public institutions and private enterprises. Governments have begun to examine its policy implications, while technology firms and platform providers are investing heavily in infrastructure, content, and services that support immersive digital experiences. Industry forecasts indicate that by 2026, a substantial proportion of the global population may regularly participate in metaverse-related activities, ranging from work and education to entertainment and social interaction [5]. At the same time, the growing integration of the metaverse into everyday life raises critical challenges related to ethics, governance, privacy, and digital inclusion [6, 7]. Addressing these issues is essential to ensure that the development of the metaverse remains socially responsible and aligned with broader public values.

The rapid evolution of the metaverse cannot be attributed to a single technological breakthrough. Instead, it emerges from the dynamic convergence of multiple advanced digital technologies, each contributing distinct capabilities to the construction of immersive and interactive virtual environments. Their coordinated integration provides the technical foundation for real-time presence, large-scale multi-user interaction, and the seamless coupling of virtual and physical experiences. Rather than functioning independently, these technologies operate as an interconnected system that continuously expands the scope and complexity of digital environments. Together, they are reshaping how individuals communicate, collaborate, and participate in social and economic activities, thereby accelerating the transition toward hybrid forms of digital life.

A critical enabler of the metaverse ecosystem is the concept of digital twins, defined as real-time, data-driven virtual representations of physical entities, processes, or systems [8]. Continuously updated through sensor networks, telemetry, and machine learning algorithms, digital twins support high-fidelity simulation, real-time

monitoring, and predictive optimization across domains such as smart manufacturing, healthcare, urban planning, and environmental management [9]. Within the metaverse, their role extends beyond static mirroring of reality. Digital twins function as interactive and adaptive components that tightly couple physical infrastructure with virtual environments. They enable responsive avatars, simulate dynamic physical processes, and support immersive experiences that evolve in parallel with real-world changes. This bidirectional integration strengthens the linkage between virtual and physical realms, allowing the metaverse to operate not only as an interaction space but also as a decision-support layer embedded within real-world systems and governance processes.

To satisfy the stringent latency and responsiveness requirements of immersive applications, edge computing shifts computational and storage resources from centralized cloud infrastructures to geographically distributed servers located closer to end users [10, 11]. By shortening data transmission paths, edge computing substantially reduces end-to-end latency, which is essential for metaverse functionalities such as real-time gesture recognition, synchronized multi-user interaction, and adaptive content generation [12]. Beyond improving responsiveness, this architectural paradigm enhances system reliability in scenarios where continuous low-latency interaction is critical. Moreover, edge computing contributes to the scalability and accessibility of metaverse services by offloading computationally intensive tasks from user devices to edge nodes. This capability lowers hardware requirements at the client side and enables a wider range of users to access high-quality immersive environments without reliance on expensive local computing resources. As a result, edge computing plays a central role in expanding the reach of the metaverse across heterogeneous devices and network conditions.

At the network layer, the emergence of sixth-generation (6G) wireless communication represents a major step toward meeting the performance demands of large-scale immersive virtual worlds. With the goal of achieving sub-millisecond latency and terabit-per-second data rates, 6G is expected to support the transmission of

high-bandwidth, multimodal sensory data and enable seamless real-time interaction among distributed users [13]. Such capabilities are critical for advanced metaverse applications involving spatial computing, haptic feedback, and tightly synchronized collaboration. In addition to improvements in speed and latency, 6G is anticipated to integrate network intelligence, enabling dynamic allocation of bandwidth, computing resources, and connectivity based on real-time application requirements. This adaptive behavior is essential for maintaining service quality across metaverse applications with diverse and competing performance demands. Furthermore, the promise of ubiquitous coverage and seamless mobility will allow users to access immersive virtual environments continuously across locations, supporting persistent and context-aware metaverse experiences [14].

Artificial Intelligence (AI) constitutes a foundational pillar of the metaverse by providing the core capabilities for perception, decision-making, and adaptation. AI technologies are embedded across multiple layers of the metaverse architecture, ranging from user-facing interaction mechanisms to backend system management [15]. Capabilities such as behavior prediction, natural language processing, semantic understanding, and real-time content generation enable personalized and context-aware experiences that evolve dynamically with user interaction. At the system level, AI supports real-time resource orchestration by adjusting allocations of computing, storage, and bandwidth in response to changing user behavior and network conditions [16]. This adaptive optimization enhances scalability and ensures consistent Quality of Experience (QoE) across heterogeneous devices and access environments. Beyond infrastructure management, AI also underpins intelligent content creation and interaction, including autonomous non-player characters, procedural environment generation, and automated moderation mechanisms that address ethical and behavioral challenges in virtual spaces.

In parallel, blockchain technology provides essential mechanisms for establishing trust, transparency, and secure value exchange within the metaverse [17, 18]. Through decentralized verification and tamper-resistant record keeping, blockchain

reduces reliance on centralized intermediaries and enhances the credibility of digital transactions. This functionality is particularly important in environments where users generate, own, and trade digital assets such as virtual real estate, digital collectibles, and intellectual property. By ensuring ownership traceability and provenance, blockchain enables interoperable and secure asset management across platforms. In addition, blockchain supports decentralized governance models through smart contracts and Decentralized Autonomous Organizations (DAOs), allowing users to participate directly in rule formation and enforcement [19]. These mechanisms promote programmable trust and collective decision-making, reinforcing the role of users as active stakeholders rather than passive participants in metaverse ecosystems.

Complementing these technologies, 3D reconstruction techniques enable the creation of high-fidelity digital representations of physical environments and objects [20]. Leveraging methods such as LiDAR scanning, photogrammetry, and neural rendering, these techniques generate virtual scenes with high geometric accuracy and visual realism. Such fidelity enhances immersion by preserving spatial coherence and environmental detail. Applications span architectural visualization, cultural heritage preservation, and virtual tourism, where authenticity and spatial continuity are particularly important. More broadly, 3D reconstruction supports the development of hybrid environments in which physical and digital spaces are seamlessly integrated, providing a realistic and navigable foundation for immersive metaverse experiences.

Positioned at the center of user interaction with the metaverse is eXtended Reality (XR), a collective term describing environments and interfaces that integrate real and virtual elements [21]. XR encompasses a spectrum of immersive technologies, including Virtual Reality (VR), Augmented Reality (AR), and Mixed Reality (MR) [22]. VR systems fully immerse users in computer-generated environments, isolating them from the physical world and enabling navigation, object manipulation, and social interaction through avatars and virtual spaces [23]. By contrast, AR enhances the physical environment by superimposing digital content onto the real world, allowing users to perceive contextual information while remaining grounded in their surround-

ings [24]. MR further extends this paradigm by enabling digital and physical objects to coexist and interact in real time, supporting more natural and responsive forms of human–machine interaction [25]. Collectively, these XR technologies redefine how individuals perceive, interpret, and engage with their environments by enabling fluid transitions between physical reality and virtual simulation. As a result, mobile XR devices such as head-mounted displays and AR glasses have emerged as the primary access points to the social metaverse, underscoring their growing role in everyday digital experiences [26].

The rapid advancement of XR is reflected not only in conceptual frameworks and software capabilities but also in the accelerated evolution of hardware devices that mediate user interaction with virtual environments. In recent years, a wide range of commercial XR headsets has entered the market, offering varying levels of immersion, interactivity, and ergonomic comfort. These devices play a critical role in shaping user experience, as hardware characteristics such as display resolution, Field of View (FOV), and refresh rate directly influence visual fidelity, motion smoothness, and system responsiveness. As user expectations for realism and immersion continue to rise, the performance demands placed on XR hardware have increased correspondingly, exposing limitations in local computing, rendering capacity, and energy efficiency. These constraints highlight the growing importance of system-level solutions that can sustain high-quality immersive experiences without overburdening end-user devices.

Table 1.1 summarizes representative XR devices developed by major industry actors, including Meta, Apple, Microsoft, and HTC. It compares key technical characteristics such as display resolution, field of view, tracking capabilities, and input modalities, providing a concise overview of the technological diversity across contemporary XR platforms [27]. This diversity reflects differing design objectives and target usage scenarios, spanning immersive entertainment, professional productivity, collaborative work, and mixed-reality interaction.

Recent devices such as the Meta Quest 3 and Apple Vision Pro illustrate a grow-

Table 1.1: Representative XR Devices and Their Specifications

Device Name	**Type**	**Year**	**Resolution**	**FOV**	**Refresh Rate**
Meta Quest 2	VR	2020	1832×1920	90°	90Hz
Meta Quest 3	MR	2023	2064×2208	110°	120Hz
Apple Vision Pro	MR	2024	3660×3200	100°	100Hz
Microsoft HoloLens 2	AR	2019	1440×936	43°	60Hz
HTC Vive Pro 2	VR	2021	2448×2448	120°	120Hz
PlayStation VR2	VR	2023	2000×2040	110°	120Hz
Pico 4	VR	2022	2160×2160	105°	90Hz
Magic Leap 2	AR	2022	1440×1760	45°	120Hz
Varjo XR-4	MR	2023	1920×1920	115°	90Hz

ing industry emphasis on MR-oriented platforms that seek to tightly integrate digital content with the physical environment. These systems are designed to support context-aware interaction, real-world spatial anchoring, and smooth transitions between immersive and augmented experiences, thereby enabling more natural and continuous forms of user engagement. In contrast, XR devices such as the HTC Vive Pro 2 and PlayStation VR2 focus on high-fidelity VR, prioritizing visual realism, wide field of view, and precise motion tracking to support demanding applications including gaming, training simulations, and 3D visualization.

These contrasting design trajectories highlight the absence of a one-size-fits-all XR solution and underscore the increasing heterogeneity of user requirements, device capabilities, and application contexts. This heterogeneity poses fundamental challenges for metaverse systems that must deliver consistent QoE across diverse hardware platforms, motivating the need for adaptive, system-level approaches to rendering, networking, and resource management.

1.2 Motivation

Despite notable progress in immersive rendering, interaction design, and network infrastructure, the majority of current metaverse platforms remain constrained by limited system autonomy and rigid operational logic. Interactive objects and virtual entities are commonly driven by predefined scripts or narrowly scoped behavioral rules, which restrict their ability to interpret contextual information, reason about evolving situations, and respond adaptively to complex user interactions. As a result, most content creation and environmental updates depend heavily on manual configuration or template-based pipelines, offering little support for continuous and autonomous evolution of virtual worlds.

In parallel, existing metaverse infrastructures face persistent challenges in efficiently coordinating computational, rendering, and communication resources under heterogeneous and rapidly changing workloads. User interaction patterns, scene complexity, and network conditions often fluctuate unpredictably, yet resource management mechanisms typically lack the flexibility to respond in real time. Consequently, although many platforms can deliver impressive visual realism under controlled conditions, they struggle to sustain long-term responsiveness, scalability, and intelligent adaptation. This gap highlights a fundamental limitation of current metaverse systems, which prioritize visual fidelity while offering limited support for persistent intelligence and autonomous system behavior.

Overcoming these constraints requires a conceptual shift toward intelligent systems capable of autonomous perception, reasoning, and action. Within this context, an AI agent is generally understood as a goal-oriented computational entity that can perceive its environment, maintain internal state, utilize external tools, and adjust its actions based on feedback over time [28]. This definition reflects a departure from isolated, prompt-driven models toward interactive systems that exhibit continuity, adaptability, and situational awareness. Recent advances in Multimodal Large Language Models (MLLMs) have significantly strengthened the capabilities of such agents

by enabling integrated reasoning across textual, visual, and other sensory modalities. These developments have enhanced performance in tasks such as visual–language grounding, multimodal inference, and tool-assisted decision-making, all of which are essential for complex virtual environments.

However, as the scope of agent responsibilities expands toward open-ended objectives and long-horizon tasks, the limitations of single-agent architectures become increasingly evident. Individual agents often struggle to maintain coherent reasoning over extended interactions, scale effectively across concurrent users, or remain robust under dynamic and uncertain conditions. In response to these challenges, the concept of agentic AI has gained prominence. Rather than emphasizing isolated intelligence, agentic AI focuses on structured reasoning, task decomposition, long-term planning, and sequential execution with minimal human supervision [29, 30]. Central to this paradigm is the adoption of coordinated multi-agent systems, in which specialized agents share contextual information, collaborate in decision-making, and execute interdependent actions. Through such coordination, agentic AI systems can address complex and evolving problems that exceed the cognitive, temporal, and operational limits of individual agents.

Embedding agentic AI into metaverse systems therefore represents a promising direction for addressing the structural limitations of existing platforms. By enabling autonomous, goal-directed behavior through coordinated multi-agent collaboration, agentic metaverse systems have the potential to improve adaptiveness, scalability, and sustained user QoE in highly interactive and dynamic environments. Nevertheless, translating this vision into practical and deployable systems introduces several fundamental challenges that remain insufficiently explored:

- Constructing a unified system architecture for agentic metaverse environments is intrinsically complex due to the heterogeneity of agent functionalities and performance requirements. Agents responsible for global reasoning, perception, control, and real-time interaction differ substantially in terms of latency

tolerance, computational intensity, and preferred execution environments. Determining how these agents should be distributed across thing, edge, and cloud layers, while maintaining efficient coordination, consistency of shared context, and timely information exchange under dynamic conditions, constitutes a core systems-level challenge. Existing designs are often narrowly tailored to specific tasks, which limits generality, cross-agent collaboration, and holistic system optimization.

- Supporting agentic behavior also places stringent demands on real-time resource management. While edge computing reduces communication latency, many current allocation strategies rely on static heuristics or predefined policies that cannot adapt effectively to variations in user behavior, network dynamics, or scene complexity. This lack of adaptability leads to inefficient bandwidth utilization and unstable performance, making it difficult to maintain consistently high QoE as workloads and interaction patterns evolve over time.

- In addition, systematic evaluation methodologies for assessing the cognitive capabilities of MLLMs within metaverse contexts remain underdeveloped. Most existing benchmarks emphasize closed-form tasks or static answer accuracy, which fail to capture the open-ended, context-sensitive reasoning required for agentic decision-making in immersive environments. As a result, the extent to which current MLLMs can provide reliable global cognition and coordination support for agentic metaverse systems remains unclear, underscoring the need for dedicated evaluation frameworks aligned with metaverse-specific perceptual and semantic demands.

1.3 Thesis Contributions

This thesis aims to address several fundamental challenges in the realization of agentic metaverse systems, spanning system architecture, resource management, and cogni-

tive evaluation. The main contributions of this thesis are summarized as follows:

- We design and implement a metaverse testing platform that supports real-time and interactive exploration of a high-fidelity three-dimensional model of New York City. This platform serves as a realistic experimental environment for investigating user interaction behavior, system workload characteristics, and network performance under immersive conditions. Through comprehensive measurements, we conduct a systematic comparison between local rendering and remote rendering strategies. Experimental results show that remote rendering delivers substantial improvements in end-to-end latency and rendering frame rate, especially in large-scale and computation-intensive scenes. These findings provide empirical evidence for the necessity and effectiveness of remote-rendered architectures in supporting scalable metaverse systems.

- We propose a unified thing–edge–cloud architecture that systematically organizes heterogeneous agents across network layers to enable coherent perception, reasoning, and action in immersive virtual environments. The proposed architecture explicitly differentiates agent roles according to their functional objectives and performance constraints, allowing latency-sensitive agents to operate at the edge while computation-intensive cognitive agents are deployed in the cloud. Through structured coordination and information exchange across layers, the architecture supports closed-loop interaction between high-level reasoning and real-time execution. This design establishes a general and extensible system foundation for agentic metaverse deployments, addressing the limitations of fragmented and task-specific architectures in existing metaverse systems.

- We propose a time-step-based QoE model tailored to metaverse environments. The model integrates key network performance indicators with user-centric experience factors, including perceived scene quality, frame choppiness, interaction latency, and metaverse frame instability. These factors have been identified in prior user studies as critical determinants of immersive experience quality. To

validate the proposed model, we conduct extensive subjective evaluations using the Mean Opinion Score (MOS) methodology. The results demonstrate that the model reliably captures user-perceived quality variations in dynamic XR scenarios and provides an effective quantitative foundation for adaptive system optimization.

- We introduce a novel resource allocation framework based on multi-agent soft actor-critic (MASAC) reinforcement learning enhanced with graph convolutional networks (GCNs). In this framework, each agent is responsible for dynamically allocating communication and computation resources for an individual metaverse user. Communication demand is represented by the bitrate of rendered scene streams, while computational demand is quantified by CPU utilization. To capture the spatially distributed and highly dynamic nature of metaverse environments, we model the system as a time-varying graph in which nodes correspond to agents and edges encode interaction influence among users. By integrating GCNs with a self-attention mechanism, the proposed framework enables agents to learn cooperative policies under non-stationary conditions, effectively adapting to user mobility and changing interaction patterns. To the best of our knowledge, this work represents the first attempt to combine SAC and GCN-based relational modeling for resource allocation in agentic metaverse systems.

- We propose a dedicated evaluation paradigm for assessing the cognitive capabilities of MLLMs in metaverse environments. Instead of relying on closed-form accuracy metrics, the proposed approach adopts an MLLM-as-a-judge framework equipped with metaverse-specific semantic criteria, enabling rubric-based and open-ended evaluation of perceptual understanding, contextual reasoning, and interpretive consistency. This paradigm is specifically designed to reflect the cognitive demands faced by agentic metaverse systems, where agents must reason about complex scenes, user intent, and evolving contexts. The proposed evaluation framework provides a principled and scalable methodology for bench-

marking MLLMs as cognitive components in agentic metaverse architectures.

1.4 Scholarly Achievements

The publications based on the research conducted in this thesis are listed below.

- Zijian Long, Haiwei Dong, Abdulmotaleb El Saddik, "Human-Centric Resource Allocation for the Metaverse with Multi-Access Edge Computing", *IEEE Internet of Things Journal*, vol. 10, no. 22, pp. 19993-20005, 2023
- Zijian Long, Haopeng Wang, Haiwei Dong, and Abdulmotaleb El Saddik, "Adaptive Social Metaverse Streaming Based on Federated Multiagent Deep Reinforcement Learning", *IEEE Transactions on Computational Social Systems*, vol. 12, no. 5, pp.3804-3815, 2025
- Zijian Long, Haiwei Dong, Abdulmotaleb El Saddik, "Interacting with New York city data by HoloLens through remote rendering", *IEEE Consumer Electronics Magazine*, vol. 11, no. 5, pp. 64-72, 2022
- Zijian Long, Rajwa Alharthi, Abdulmotaleb El Saddik, "Needfull–A tweet analysis platform to study human needs during the COVID-19 pandemic in new york state", *IEEE Access*, vol. 8, pp. 136046-136055, 2020
- Haopeng Wang, Zijian Long, Haiwei Dong, and Abdulmotaleb El Saddik, "MADRL-Based Rate Adaptation for 360° Video Streaming With Multi-Viewpoint Prediction", *IEEE Internet of Things Journal*, vol. 11, no. 15, pp. 26503-26517, 2024.
- Zijian Long, Haiwei Dong, Abdulmotaleb El Saddik, "A Collaborative Thing-Edge-Cloud Architecture for Agentic AI-Enabled Metaverse", *IEEE Internet of Things Journal* (submitted)

1.5 Thesis Outline

The remainder of this thesis is organized as follows:

- Chapter 2 provides background and a comprehensive review of related work. It surveys the evolution of the metaverse and its enabling technologies, with particular attention to adaptive bitrate control and resource allocation strategies in edge computing environments. The chapter further reviews the theoretical foundations and recent advances in multi-agent deep reinforcement learning (MADRL) and visual scene understanding.

- Chapter 3 investigates network characteristics and user interaction patterns in metaverse scenarios through a practical case study based on a high-fidelity three-dimensional model of New York City. Using immersive XR devices, the chapter conducts a systematic comparison between remote rendering and local rendering approaches. Performance differences are analyzed across key experience-related metrics, including end-to-end latency, rendering frame rate, and system stability, providing empirical motivation for remote-rendered and edge-assisted metaverse architectures.

- Chapter 4 presents a unified thing–edge–cloud architecture for agentic metaverse systems. The chapter details the roles and interactions of heterogeneous agents deployed across different layers, and explains how perception, reasoning, and action are coordinated within the proposed framework. Core system components, communication mechanisms, and key performance metrics are introduced to illustrate how the architecture supports scalable, adaptive, and coherent agentic behavior.

- Chapter 5 introduces the proposed edge resource allocation framework based on MASAC reinforcement learning. The chapter formulates the resource allocation problem under dynamic metaverse workloads and describes the integration

of graph convolutional networks and self-attention mechanisms. Extensive experiments are conducted under diverse network conditions and user interaction scenarios to evaluate the effectiveness of the proposed method in improving resource efficiency and user QoE.

- Chapter 6 presents a dedicated benchmarking framework for evaluating the cognitive capabilities of multiple representative MLLMs in metaverse environments. The chapter introduces metaverse-specific evaluation criteria and an MLLM-as-a-judge paradigm for rubric-based, open-ended assessment. Experimental results and human validation are reported to analyze the strengths and limitations of current MLLMs as cognitive components in agentic metaverse systems.

- Chapter 7 concludes the thesis by summarizing the main findings and contributions. It discusses the limitations of the current work and outlines promising directions for future research on agentic metaverse systems, including architectural extensions, learning algorithms, and cognitive evaluation methodologies.

Chapter 2

Literature Review

This chapter presents a comprehensive and structured review of five closely related research domains that collectively underpin the development of adaptive and intelligent streaming systems for the metaverse. These domains span both foundational theories and recent state-of-the-art advances at the intersection of immersive media systems, intelligent network control, and distributed ML. Rather than treating these strands in isolation, this review emphasizes their complementary roles in addressing the unique challenges of metaverse environments, including ultra-high-quality immersive rendering, stringent latency constraints, dynamic network conditions, and heterogeneous computing resources.

Specifically, the chapter examines: (1) the evolution of the metaverse and its enabling technologies, which establish the system-level requirements and architectural context; (2) learning-based ABR control strategies, which form the core of client-side streaming intelligence; (3) resource allocation mechanisms in edge computing, which provide low-latency computational and networking support; (4) MADRL approaches for decentralized optimization under multi-user and multi-resource settings; and (5) visual scene understanding techniques, which enable semantic-aware perception and decision-making in immersive virtual environments. Together, these research streams form an integrated conceptual and technical foundation for the agentic and adaptive metaverse systems investigated in this thesis.

2.1 Development of the Metaverse

The development of the metaverse can be conceptualized as a multi-stage evolutionary process characterized by progressive changes in technological capabilities, modes of user interaction, and socio-economic functions. Rather than emerging as a discrete technological breakthrough, the metaverse has evolved through the gradual convergence of immersive media technologies, network infrastructures, and participatory digital cultures [31, 32]. This section reviews the historical trajectory of the metaverse across four major stages, highlighting how it has transformed from speculative imagination into a complex sociotechnical system embedded in contemporary digital life.

- Conceptual Origins and Early Prototypes (1960s–2000s):

  The earliest conceptual foundations of the metaverse can be traced to speculative fiction and pioneering research on digital communication. Long before the term itself was formalized, visions of shared virtual spaces in which users could inhabit persistent digital identities appeared in influential literary works. Neal Stephenson's novel Snow Crash articulated an early conceptualization of a persistent, avatar-based virtual world structured around social interaction and digital economies. Alongside these fictional narratives, early virtual platforms such as Second Life [33] and The Sims Online [34] emerged during the late 1990s and early 2000s as practical experiments in large-scale online social environments. Although constrained by limited bandwidth, rudimentary graphics, and modest computational resources, these systems introduced several defining characteristics of the metaverse, including user-generated content, persistent virtual spaces, and digitally mediated social interaction. Advances during this period in Internet connectivity, graphical user interfaces, and early virtual reality technologies established the technical and conceptual foundations for subsequent developments.

- Infrastructure Expansion and Immersive Technology Growth (2000s–2015):

  The second stage was marked by significant expansion of the technological infrastructure required to support immersive and interactive virtual environments. The widespread adoption of broadband Internet and mobile networking enabled real-time communication and high-bandwidth content delivery across geographically distributed users. In parallel, cloud computing emerged as a critical enabling technology by providing scalable and elastic computational resources for large virtual platforms. At the device level, consumer-oriented immersive hardware such as Oculus Rift, HTC Vive, and Microsoft HoloLens substantially improved display resolution, motion tracking accuracy, and interaction fidelity, leading to deeper user immersion. Massively multiplayer online games, exemplified by World of Warcraft [35], demonstrated the feasibility of maintaining persistent virtual worlds with complex social structures and in-game economies. Beyond entertainment, this period also saw early applications of immersive technologies in domains such as medical training, architectural visualization, and distance education, signaling the expanding functional scope of metaverse-related systems.

- Platform Convergence and the Rise of Virtual Economies (2015–2020):

  The third phase was characterized by increasing platform convergence, enhanced interoperability, and the formation of structured virtual economies. Improvements in software frameworks and cross-platform development tools enabled users to access immersive experiences through heterogeneous devices, including personal computers, mobile terminals, and XR headsets. During this period, blockchain technologies began to play a prominent role in virtual environments by supporting decentralized ownership and secure digital transactions. The emergence of non-fungible tokens, as reviewed in [36], enabled the authentication, exchange, and monetization of unique virtual assets. Platforms such as Fortnite [37], Minecraft VR, and Roblox VR facilitated ecosystems in which users could simultaneously act as consumers, creators, and economic partici-

pants. As a result, the metaverse increasingly functioned as a socio-economic space that supported digital labor, virtual commerce, and cultural production alongside entertainment and social interaction.

- Toward an Integrated and Persistent Metaverse (2020s–present):

  In its most recent stage, the metaverse has evolved toward a more integrated and persistent digital infrastructure that increasingly overlaps with everyday human activities. Its application domains now extend across entertainment, education, healthcare, collaborative work, and social communication. Advances in artificial intelligence, low-latency 5G and emerging 6G networks, decentralized application frameworks, and increasingly sophisticated XR interfaces have jointly enhanced immersion, personalization, and system scalability. Contemporary platforms such as Meta Horizon Worlds, Microsoft Mesh, and decentralized virtual worlds including Decentraland [38] exemplify this stage of development. These systems support persistent digital identities, cross-platform continuity, and hybrid physical-digital interaction, thereby reducing the separation between virtual and physical environments. At this stage, the metaverse is best understood not as a single platform but as a distributed and evolving infrastructure that enables large-scale social interaction, collaborative productivity, and economic exchange in immersive digital spaces.

Across these successive stages, the metaverse has progressed from abstract conceptualization to an operational yet continuously evolving sociotechnical system. Its development reflects not only advances in immersive and networked technologies but also broader transformations in how individuals interact with information, environments, and one another within digital contexts.

2.2 Learning-Based Adaptive Bit Rate Control

Adaptive Bit Rate (ABR) control constitutes a fundamental component of modern video streaming systems, enabling dynamic adjustment of encoding bitrate in response to time-varying network conditions. Its primary objective is to balance uninterrupted playback and visual quality so as to maximize end-user QoE [39]. Conventional ABR schemes typically rely on handcrafted heuristics based on buffer occupancy, throughput estimation, or predefined adaptation rules, and have been widely adopted in streaming standards such as MPEG-DASH and HTTP Live Streaming (HLS). While effective in relatively stable environments, these reactive strategies often exhibit suboptimal performance under highly dynamic or latency-sensitive scenarios, including cloud gaming and immersive virtual environments. These limitations have motivated a transition toward learning-based ABR control, in which Deep Reinforcement Learning (DRL) is employed to autonomously derive bitrate adaptation policies through sequential interaction with the network environment. By formulating ABR as a Markov Decision Process (MDP), DRL-based methods enable adaptive and forward-looking decision-making under uncertain and non-stationary conditions.

A representative milestone in this research direction is Pensieve, proposed by Mao et al. [40], which applies an actor–critic DRL framework to learn ABR policies directly from real-world network traces. Pensieve demonstrated consistent improvements over rule-based baselines across both simulated environments and practical deployments, establishing DRL as a viable alternative to heuristic-driven ABR control. Building on this foundation, subsequent studies have focused on improving robustness and generalization. For instance, Huang et al. proposed Tiyuntsong, which integrates adversarial training into the learning process to expose the ABR agent to a wider range of network dynamics, thereby enhancing policy resilience under severe bandwidth fluctuations and delay variations [41].

The emergence of real-time interactive services has further intensified research interest in DRL-based ABR control. In applications such as cloud gaming and XR

streaming, stringent latency constraints and rapid viewpoint changes impose additional challenges beyond conventional video-on-demand scenarios. Chen et al. introduced a DRL-driven congestion control mechanism within the WebRTC framework, replacing the traditional Google Congestion Control (GCC) algorithm with an Asynchronous Advantage Actor–Critic (A3C) model to jointly adapt bitrate and frame rate under fluctuating bandwidth and delay conditions [42]. More recently, Wei et al. developed a transformer-based DRL architecture for cloud XR streaming, leveraging temporal attention mechanisms and multi-agent coordination to improve bitrate adaptation across heterogeneous content streams and user demands [43].

As immersive applications evolve toward multi-user and shared virtual environments, such as the metaverse, ABR control increasingly operates in settings characterized by resource contention, user interaction, and coupled decision processes. Single-agent learning paradigms are therefore insufficient to capture the interdependencies among concurrent users and shared network or rendering resources. To address this challenge, Wang et al. proposed an immersive streaming control framework that combines multi-agent DRL with edge intelligence to jointly schedule bitrate selection and processing resources across distributed rendering nodes [44]. Complementing these system-level contributions, Subhan et al. provided a comprehensive survey demonstrating the growing importance of DRL and multi-agent DRL techniques for media delivery in 5G and emerging 6G networks, particularly in scenarios requiring simultaneous support for low latency and ultra-high bandwidth [45].

Collectively, these studies illustrate a clear evolution from heuristic-driven ABR mechanisms toward intelligent, data-driven control systems capable of both local adaptation and coordinated optimization in dynamic and distributed media networks. As immersive and user-driven services continue to proliferate, future ABR solutions are expected to further incorporate cooperative learning, context awareness, and distributed intelligence. These trends motivate the integration of ABR control with edge computing and multi-agent optimization frameworks, which are examined in the following sections.

2.3 Resource Allocation for Edge Computing

Edge computing has emerged as a foundational architectural paradigm for supporting ultra-reliable, low-latency, and bandwidth-intensive services in next-generation digital environments. By deploying computational, storage, and networking resources in close proximity to end users, typically at the edge of radio access networks, edge computing significantly reduces backhaul dependency and service latency while improving real-time responsiveness [46, 47]. These characteristics make edge computing particularly well suited for metaverse systems, which demand high concurrency, low-latency interaction, and continuous synchronization across distributed users. Representative workloads in this context include real-time XR streaming, avatar interaction, collaborative scene construction, and volumetric media processing. Effective support for such services requires intelligent resource orchestration mechanisms capable of dynamically allocating heterogeneous resources, including CPU cores, GPU accelerators, memory, storage, and wireless spectrum, in response to rapidly varying service demands [48, 49]. Unlike traditional mobile applications with relatively stable traffic patterns, metaverse platforms exhibit temporally bursty and spatially heterogeneous workloads, motivating the adoption of adaptive and learning-driven resource management strategies.

Among existing approaches, DRL has attracted significant attention due to its ability to learn control policies in dynamic, high-dimensional environments. In DRL-based edge computing resource allocation, the system is typically modeled as a sequential decision-making process in which an agent, often implemented at the edge computing controller or orchestrator, observes the system state and selects resource allocation actions to optimize long-term performance objectives such as latency reduction, throughput maximization, or balanced resource utilization. A representative example is the DRLRA framework proposed by Wang et al. [50], which employs a Deep Q-Network (DQN) to manage computation offloading and resource scheduling in a Software-Defined Networking (SDN)-enabled edge computing architecture.

By continuously monitoring traffic demands and server workloads, the agent learns adaptive policies that minimize service delay and balance computational load across distributed edge nodes. The integration of SDN provides a global network view, enhancing situational awareness and facilitating informed decision-making.

Building on centralized learning paradigms, subsequent research has explored decentralized and distributed DRL approaches to improve scalability and responsiveness. Liu et al. proposed a multi-agent DQN-based framework for decentralized task offloading and joint resource allocation [51]. In this model, individual user devices act as autonomous agents that independently learn offloading strategies based on local observations, energy constraints, and perceived competition for shared edge resources. Through decentralized learning and limited coordination, the system achieves reduced task latency and energy consumption without relying on a centralized controller, making it suitable for highly dynamic and large-scale deployments.

More recent studies have extended these ideas to metaverse-oriented networks characterized by multi-layered infrastructure and complex service interactions. Hevesli et al. introduced a multi-agent DRL framework for queue-aware task offloading in hierarchical edge computing-enabled air-ground networks [52]. Their architecture comprises three types of agents, corresponding to ground-based edge computing servers, unmanned aerial vehicle edge nodes, and user terminals, each learning coordinated policies tailored to their functional roles. This hierarchical design enables dynamic workload migration across infrastructure layers based on real-time queue states and user mobility, thereby maintaining ultra-low-latency service continuity for immersive metaverse applications.

In a complementary direction, Chu et al. proposed a DRL-based multi-tier resource allocation framework that supports hierarchical decision-making across multiple edge layers in metaverse environments [53]. Their approach integrates spectrum allocation, GPU scheduling, and memory management into a unified control loop driven by actor–critic reinforcement learning. Such joint optimization enables efficient support for compute-intensive and bandwidth-demanding services, including

multi-user XR collaboration and digital twin rendering. Beyond traditional DQN and A3C-based methods, emerging research also explores context-aware DRL strategies in which resource allocation policies adapt based on inferred user intent, content characteristics, or service-level requirements. For instance, Yu et al. developed a DRL-enhanced edge computing scheduler that prioritizes tasks according to virtual user behavior patterns extracted using knowledge graphs and temporal embeddings [54]. By incorporating semantic and behavioral context, these approaches enable proactive scheduling and more precise alignment between edge resource provisioning and immersive service demands.

Overall, learning-based edge computing resource allocation has evolved from centralized control toward distributed, hierarchical, and context-aware frameworks. This evolution reflects the growing complexity of metaverse workloads and the need for scalable orchestration mechanisms that can sustain interactive, high-performance services under dynamic conditions. These developments naturally motivate the adoption of multi-agent reinforcement learning paradigms, which are discussed in the following section.

2.4 Multi-Agent Deep Reinforcement Learning

MADRL extends classical reinforcement learning to environments in which multiple autonomous agents interact concurrently with both the environment and one another. In contrast to single-agent settings, where the environment is typically assumed to be stationary and independent of the agent's actions, multi-agent environments are inherently dynamic and non-stationary. The learning policy of each agent continuously influences the state transitions and reward structures experienced by others, significantly increasing the complexity of policy optimization. These challenges are further amplified in decentralized scenarios characterized by partial observability, limited communication, and distributed control, all of which are common in large-scale cyber-physical systems.

Owing to its flexibility and scalability, MADRL has emerged as a powerful framework for addressing high-dimensional decision-making problems that are difficult to solve using conventional optimization methods. By allowing agents to learn adaptive, cooperative, or competitive behaviors through interaction, MADRL has been successfully applied to domains such as traffic signal control, cooperative robotics, smart grid operation, and wireless resource management. These application domains share fundamental structural characteristics with metaverse systems, including distributed decision-making, shared resource constraints, and stringent real-time requirements. As a result, MADRL is particularly well suited for coordinating heterogeneous agents in metaverse environments, where users, edge servers, and network components must operate collaboratively under dynamic and uncertain conditions.

A central methodological challenge in MADRL lies in mitigating the non-stationarity induced by simultaneous policy updates across agents. Without appropriate mechanisms, this non-stationarity can lead to unstable training dynamics and poor convergence. To address this issue, Lowe et al. introduced the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm [55], which adopts a centralized training and decentralized execution (CTDE) paradigm. During training, each agent is equipped with a centralized critic that has access to the joint state and action information of all agents, enabling more stable policy learning. During execution, however, agents rely solely on local observations and decentralized policies, ensuring scalability and feasibility in distributed environments. This separation between training and execution is particularly advantageous for edge-enabled metaverse systems, where global information may be available offline or during training but not at runtime.

Building upon the CTDE framework, subsequent research has explored structured information sharing and relational reasoning among agents. Jiang et al. proposed a graph-based MADRL approach that integrates GCNs into the learning process [56]. In this framework, agents are represented as nodes in a graph, while edges encode spatial, logical, or functional relationships. GCNs enable each agent to aggregate

information from neighboring agents, facilitating localized coordination without requiring full global observability. Such graph-based representations are well aligned with metaverse scenarios, where agents are distributed across physical and virtual spaces but remain interconnected through network and interaction graphs.

Scalability remains a critical concern in MADRL, particularly in environments involving a large number of interacting agents. To address the combinatorial explosion of agent interactions, Yang et al. introduced the Mean Field Reinforcement Learning (MFRL) framework [57]. Instead of explicitly modeling pairwise interactions among agents, MFRL approximates the collective influence of other agents using a mean field representation. This approximation significantly reduces the dimensionality of the learning problem while preserving essential interaction dynamics, making it suitable for large-scale systems in which hundreds or thousands of agents operate concurrently. In metaverse environments, such agents may include user avatars, edge computing nodes, and intelligent network elements, all of which require coordinated yet scalable control.

Despite these advances, several challenges remain in deploying MADRL within real-world, edge-based metaverse systems. Communication overhead among distributed agents can become prohibitive under bandwidth-constrained or unreliable wireless conditions. Ensuring stable convergence in non-stationary environments with delayed or incomplete feedback remains an open research problem. In addition, agent heterogeneity, arising from differences in computational capability, network connectivity, and sensing accuracy, complicates policy generalization and robustness. Finally, the strict real-time requirements of immersive metaverse applications necessitate lightweight and computationally efficient learning algorithms that can be deployed on resource-constrained hardware platforms.

2.4.1 Visual Scene Understanding

Visual scene understanding concerns the capability of intelligent systems to transform raw visual observations into structured semantic representations that capture objects, spatial configurations, interactions, and overall contextual meaning. Unlike low-level perception tasks that focus on pixel-level or object-level recognition, scene understanding emphasizes holistic interpretation, requiring models to reason about relationships, spatial layouts, and latent semantics embedded in visual data. Early research in this area primarily relied on task-specific computer vision pipelines, where convolutional neural networks were applied to problems such as image classification, object detection, and semantic segmentation. Architectures including ResNet and subsequent transformer-based vision models, such as Vision Transformer and DETR, achieved substantial progress in static visual recognition by improving feature representation and global context modeling [58, 59]. However, these approaches largely operated in isolation from language and reasoning capabilities, limiting their ability to support higher-level semantic inference.

Recent advances in multimodal learning have fundamentally reshaped visual scene understanding by tightly coupling visual encoders with large language models. This integration enables systems to map visual observations into a shared vision–language representation space, supporting open-ended reasoning tasks such as image captioning, visual question answering, and spatial or relational inference. By leveraging the linguistic prior and reasoning capacity embedded in LLMs, multimodal large language models are able to move beyond rigid label prediction toward flexible semantic interpretation. As a result, scene understanding has evolved from static recognition toward descriptive, explanatory, and context-aware perception that can generalize across tasks and domains.

Beyond static imagery, emerging research increasingly focuses on dynamic and embodied environments, where visual perception unfolds over time and across viewpoints. In such settings, understanding requires reasoning over temporal sequences,

motion patterns, and three-dimensional spatial structure. To address these challenges, recent models incorporate video representations, temporal attention mechanisms, and geometry-aware encodings. For example, Video-3D LLM introduces position-aware video representations to enhance three-dimensional scene reasoning from dynamic visual inputs, while MetaVQA proposes an embodied evaluation framework that emphasizes spatial relations, agent-centric perspectives, and multi-frame contextual consistency [60, 61]. These approaches reflect a broader shift toward perception systems that are not only visually grounded but also temporally coherent and spatially aware, aligning closely with the requirements of interactive and immersive environments.

In the context of the metaverse, visual scene understanding plays a particularly critical role. Immersive virtual environments are characterized by continuous interaction, multi-user dynamics, and rapid scene changes driven by user behavior. Intelligent agents operating within such environments must not only perceive visual content accurately but also interpret semantic cues to support downstream decision-making, such as adaptive rendering, bandwidth allocation, and resource scheduling. Consequently, scene understanding becomes a foundational component for enabling semantic-aware control and agentic behavior in metaverse systems, bridging perception and action within a unified cognitive loop.

Evaluating the open-ended scene understanding capabilities of multimodal large language models presents a significant methodological challenge. Conventional automatic evaluation metrics, including n-gram overlap scores or fixed-answer accuracy, are poorly suited to free-form vision–language outputs, as they fail to capture semantic correctness, relational coherence, and contextual adequacy. These limitations are particularly pronounced in complex scene description and reasoning tasks, where multiple responses may be equally valid yet lexically diverse. To address this gap, recent studies have proposed the MLLM-as-a-judge paradigm, in which multimodal language models themselves are employed as evaluators of generated outputs [62]. By leveraging their advanced reasoning and semantic understanding capabilities, MLLM judges can perform comparative or pairwise evaluations that emphasize holistic mean-

ing rather than surface-level textual similarity. Empirical evidence suggests that this paradigm yields more robust and interpretable assessments for multimodal reasoning tasks, making it especially suitable for benchmarking scene understanding in dynamic and interactive environments.

Chapter 3

Comparative Analysis of Remote and Local Rendering in the Metaverse: A Case Study with HoloLens 2 and New York City Data

3.1 A Brief Introduction to HoloLens 2

In recent years, extended reality technologies have undergone rapid development, driven by advances in hardware capabilities and growing demand for immersive applications in domains such as healthcare, education, industrial training, and the metaverse. Within this evolving landscape, Microsoft's HoloLens 2 has emerged as a representative mixed reality headset, marking a substantial technological progression compared with its predecessor, the HoloLens 1 [63].

Relative to earlier-generation mixed reality devices, HoloLens 2 provides a noticeably enhanced level of immersion. The device offers an expanded field of view of 52 degrees and supports a higher per-eye resolution of 2048×1080 pixels, which together contribute to improved visual fidelity and spatial awareness. These enhancements enable users to perceive virtual content with greater clarity and to interact with digital objects in a manner that is more consistent with natural human perception in

physical environments.

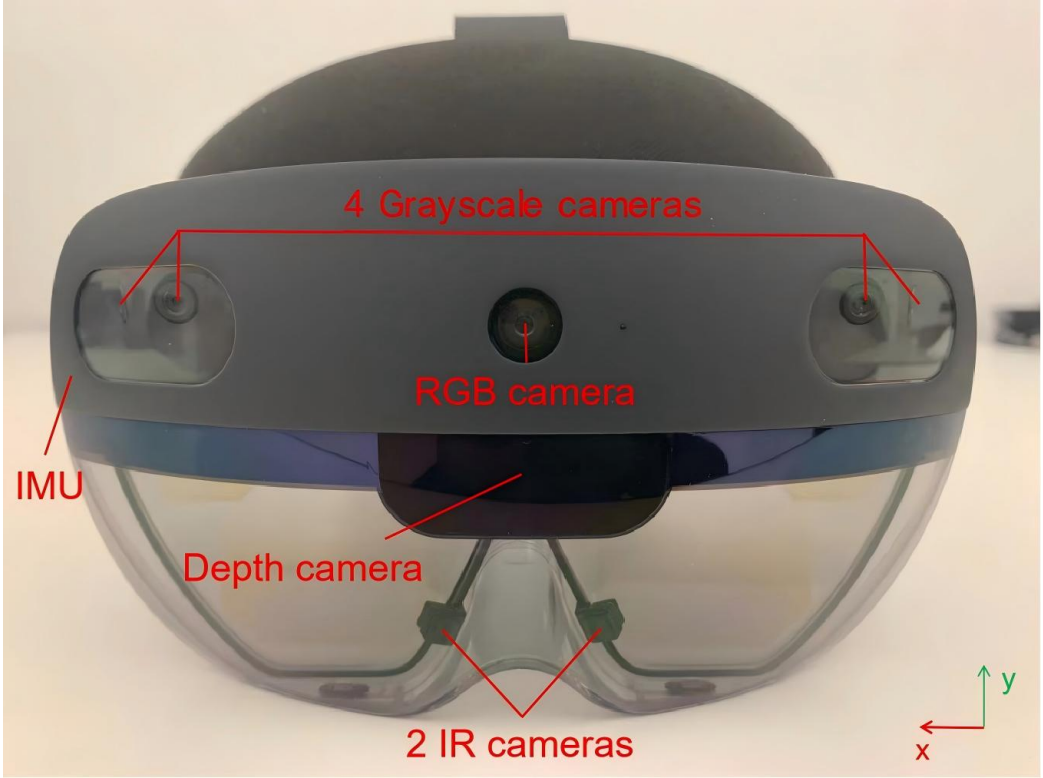


Figure 3.1: HoloLens 2 is equipped with multiple sensors to support hand tracking, eye tracking, and voice commands for human interaction. It also enables environmental understanding through six-degree-of-freedom tracking and spatial mapping.

As illustrated in Fig. 3.1, HoloLens 2 integrates a comprehensive sensor suite designed to support rich human–environment interaction. The system incorporates eight onboard cameras. Among them, four grayscale tracking cameras are dedicated to real-time visual-inertial simultaneous localization and mapping, enabling accurate and robust head pose estimation. A time-of-flight depth camera supports articulated hand tracking and three-dimensional spatial reconstruction, facilitating intuitive gesture-based interaction. In addition, two Infrared (IR) cameras are employed for precise eye-tracking, while an RGB camera is used to capture mixed reality images and video streams.

In addition to its camera system, HoloLens 2 is equipped with multiple inertial

Table 3.1: Technical specifications of cameras that the Microsoft HoloLens 2 are equipped with. These cameras introduce a large number of data for the HoloLens 2 to process to support these functions.

	Cameras	FPS	Resolution	Format	Raw data
Head Tracking	4 Grayscale	30	640 x 480	8-bit	294.9 Mbits
Hand Tracking	1 Depth	45	512 x 512	16-bit	188.7 Mbits
Eye Tracking	2 Infrared	-	-	-	-
Spatial Awareness	1 Depth	1-5	320 x 288	16-bit	1.5-7.3 Mbits

sensors to further enhance motion sensing and tracking accuracy. These include an accelerometer for measuring linear acceleration along three orthogonal axes, a gyroscope for capturing rotational motion, and a magnetometer that provides an absolute orientation reference relative to the Earth's magnetic field. Through tightly coupled sensor fusion, these components collectively enable stable tracking, low-latency interaction, and consistent spatial alignment between virtual and physical content.

The integration of high-resolution and high-frequency sensors inevitably results in substantial data generation. A representative example is the four grayscale cameras used for head tracking, which together produce approximately 294.9 Mbits/s of raw data at a frame rate of 30 frames per second, with each frame having a resolution of 640×480 pixels and 8-bit grayscale depth. As summarized in Table 3.1, the aggregate data throughput introduces significant challenges for both real-time data transmission and onboard computation.

To address these challenges, HoloLens 2 adopts a heterogeneous computing architecture that includes a custom-designed holographic processing unit alongside conventional CPU and GPU components. The holographic processing unit is specifically optimized for parallel sensor processing, spatial mapping, and environmental understanding tasks, thereby offloading computational burdens from the main processor. This design enables efficient handling of high-bandwidth sensor data while maintaining system responsiveness, exemplifying the close integration of sensing, perception,

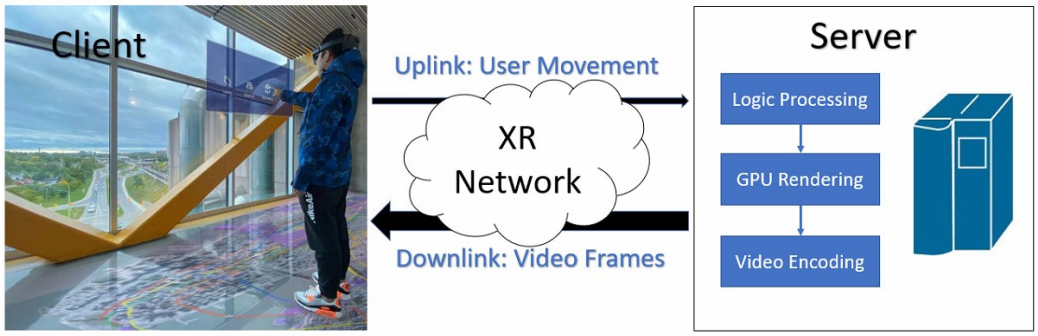


Figure 3.2: The architecture of our proposed system consists of a user with the HoloLens 2, an network, and a rendering server. On the left, a user is wearing the HoloLens 2 to interact with the New York City data.

and computation required by advanced mixed reality platforms.

Although HoloLens 2 integrates a dedicated holographic processing unit designed to support real-time spatial mapping, hand tracking, and visual localization, its on-board computational resources remain constrained when handling highly complex or data-intensive scenes. In application scenarios that demand fine-grained graphical fidelity, such as detailed inspection of mechanical assemblies including truck engines or interactive exploration of large-scale urban digital twins, local rendering on HoloLens 2 may lead to degraded performance, increased latency, or reduced visual quality.

To address these limitations, remote rendering, also referred to as cloud-based or edge-based XR rendering, has become an increasingly adopted solution. Under this paradigm, computationally intensive rendering tasks are offloaded from the HoloLens 2 to a remote server equipped with substantially more powerful hardware resources, including high-performance GPUs and large memory capacities. By transferring the rendering workload to external infrastructure, the headset can operate with a lighter computational burden while still supporting high-resolution graphics and smooth interactive experiences.

Existing empirical studies indicate that remote rendering configurations generally provide superior user QoE compared with purely local rendering, particularly under conditions involving high rendering complexity and stringent visual requirements [64].

This improvement can be attributed to the server's capability to generate high-fidelity frames at stable frame rates and to deliver them efficiently to the user device.

Figure 3.2 presents the overall architecture of the XR system based on remote rendering. In this framework, a user wearing the HoloLens 2 interacts with immersive three-dimensional digital environments. User motion, head orientation, and gesture information are captured by the headset's onboard sensors and transmitted to the remote rendering server through the network uplink. Upon receiving these inputs, the server executes application logic and renders video frames corresponding to the user's current viewpoint and interaction context. The rendered frames are then encoded and transmitted back to the HoloLens 2 via the network downlink, where they are decoded and displayed in real time.

This collaborative rendering architecture enables resource-constrained wearable devices to support visually complex XR scenes that would otherwise exceed their local processing capabilities. At the same time, it introduces new challenges related to network latency, bandwidth variability, and synchronization between user input and visual feedback. In particular, minimizing end-to-end latency is essential for preserving immersion and avoiding discomfort such as motion sickness in interactive mixed reality applications.

3.2 System Setup and Data Acquisition

To evaluate the feasibility of remote rendering for immersive XR applications, a prototype system was constructed consisting of a high-performance rendering server and a wearable client device. The rendering server operates on the Windows 10 operating system and is equipped with an Intel Core i9-11900F processor, 64 GB of DDR4 memory, and an NVIDIA GeForce RTX 3090 graphics card. This configuration provides ample computational capacity and memory bandwidth to support real-time rendering of high-resolution and large-scale three-dimensional scenes.

On the client side, the Microsoft HoloLens 2 is employed as the wearable XR

device. The headset is built on the Qualcomm Snapdragon 850 Compute Platform, integrating a multi-core CPU and GPU with 4 GB of LPDDR4x system memory. While this hardware configuration is sufficient for real-time tracking, spatial mapping, and natural interaction, the limited onboard computing resources make the device less suitable for rendering scenes characterized by dense geometry and high-resolution textures. As a result, within the proposed remote rendering architecture, the HoloLens 2 primarily functions as an interaction and display terminal rather than a full rendering platform.

System integration is realized through the Holographic Remoting Player, a software utility provided by Microsoft that enables holographic content to be streamed from a remote rendering server to the HoloLens 2. In the proof-of-concept implementation, the headset is connected to the rendering server via a USB-C cable. This wired connection ensures low-latency and high-throughput data transmission, while also reducing variability introduced by wireless communication during experimental evaluation. Although the current setup adopts a wired link for stability and repeatability, the architecture can be extended to wireless deployments using Wi-Fi or cellular networks such as 5G.

The rendering application is developed using Unity, a widely used game engine that provides comprehensive support for XR application development. To ensure cross-platform compatibility and simplify interaction with XR hardware, the OpenXR standard is adopted as the underlying application programming interface. OpenXR is an open and royalty-free specification that unifies access to XR devices and runtimes, allowing applications to be deployed across heterogeneous hardware platforms with minimal modification. Its widespread adoption by major industry stakeholders facilitates interoperability and reduces development complexity.

As a representative case study, a large-scale three-dimensional urban dataset of New York City is selected for system evaluation. The dataset is obtained from the ArcGIS Living Atlas of the World [65], a well-established repository of authoritative geospatial information. The platform provides rich urban data in the form of layered

content, including building models, terrain information, and infrastructure layouts. These information layers can be imported into Unity through compatible geographic information system toolkits and are used in this work to reconstruct a detailed virtual representation of the city, serving as a realistic and demanding test environment for immersive XR interaction.

To improve usability and support natural interaction within large-scale virtual environments, a multimodal interaction system is implemented. The system integrates three complementary interaction modalities: hand tracking, eye tracking, and voice commands, all of which are natively supported by the HoloLens 2 platform [66]. These modalities are designed to align with natural human behaviors, thereby reducing user cognitive load and eliminating reliance on handheld controllers or external input devices. By leveraging the headset's embedded sensors and real-time tracking capabilities, the proposed interaction framework enables intuitive and context-aware manipulation of holographic content within the spatial environment, allowing users with varying levels of XR experience to engage efficiently with complex urban-scale digital scenes.

3.3 Interaction Design

A key capability of the HoloLens 2 is its support for fully articulated hand tracking, which allows the system to capture and interpret fine-grained movements of the user's hands and fingers in three-dimensional space. Building on this capability, our system implements two complementary hand-based interaction techniques, namely near interaction and far interaction, which are selected according to the spatial distance between the user and the target holographic object.

For holograms located within arm's reach, users interact through direct manipulation using natural hand gestures. Operations such as pressing virtual buttons, rotating models, or dragging components are performed through contact-based gestures that closely resemble interactions with physical objects. This interaction mode

supports precise control of nearby virtual content and provides a strong sense of realism by aligning with users' everyday motor behaviors.

For holographic objects positioned beyond the user's immediate reach, a ray-based interaction technique known as point and commit is employed. This method is widely adopted in XR systems to enable efficient manipulation of distant objects without requiring physical movement. The interaction process consists of two sequential stages, as illustrated in Fig. 3.3.

- In the pointing stage, the user extends their hand, and a dashed virtual ray is projected from the palm into the scene, as shown in Fig. 3.3(a). The intersection between the ray and a holographic object provides a visual indication of the current target.
- In the committing stage, the user confirms the selection by performing a pinch gesture with the thumb and index finger. Upon this action, the dashed ray transitions into a solid line, as depicted in Fig. 3.3(b), signaling the system to execute the intended operation, such as selecting, moving, or activating the target object.

This raycasting-based interaction mechanism enables accurate manipulation of remote objects while minimizing physical effort, thereby improving user comfort and extending interaction range in large-scale XR environments.

In addition to hand-based interaction, the system exploits the eye-tracking capabilities of the HoloLens 2 to support attention-driven and hands-free interaction. The headset is equipped with inward-facing infrared cameras that continuously track eye movements and estimate the user's gaze direction, which serves as an effective proxy for visual attention and user intent.

Gaze-based interaction is particularly valuable in situations where manual input is inconvenient or when rapid, low-effort interaction is desired. In the proposed system, gaze tracking is primarily used for implicit selection and context-aware information

(a) Pointing stage (b) Committing stage

Figure 3.3: Far interaction based on hand tracking consists of two stages: a pointing stage to indicate the target object and a committing stage to confirm and execute the interaction.

presentation. For example, when a user looks at a prominent landmark in the virtual reconstruction of New York City, such as the Empire State Building or the Statue of Liberty, the system automatically displays a floating label indicating the name of the building. This mechanism enhances spatial understanding while providing immediate feedback that the system has correctly interpreted the user's focus.

To further enrich hands-free interaction, voice command recognition is integrated as a complementary modality. Voice input is captured through the built-in microphone array of the HoloLens 2 and processed by its natural language processing engine, enabling users to issue verbal commands to control system functions. In many cases, gaze and voice are combined into a multimodal interaction pipeline in which gaze specifies the target and voice determines the action. For instance, when a user fixates on a holographic button and utters the command "Open," the corresponding operation is triggered. Upon successful recognition, the system renders the recognized command text in the user's field of view for a short duration to confirm execution and reduce ambiguity.

Beyond discrete commands, voice interaction also supports navigation within the virtual city environment. By speaking the name of a well-known location, such as

"Times Square," users can prompt the system to adjust the viewpoint and transition directly to the specified area within the three-dimensional city model. This interaction paradigm facilitates efficient exploration of large-scale environments without reliance on physical controllers or complex menu structures.

Collectively, the integration of hand tracking, eye tracking, and voice commands results in a multimodal interaction framework that enhances usability, accessibility, and immersion. By leveraging natural human behaviors and minimizing the need for external input devices, the proposed design enables intuitive and non-intrusive interaction with complex XR content in large-scale virtual environments.

3.4 Visualization Results

To illustrate the effectiveness and interactivity of the proposed XR system, a comprehensive visualization interface is developed that integrates multiple information-rich components. As shown in Fig. 3.4, the user interface is organized around a central menu comprising three primary functional sections: "Base Maps", "Info Layers", and "Sightseeing". Together, these modules provide users with flexible and intuitive access to diverse forms of spatial data and urban visualizations within the three-dimensional model of New York City.

The "Base Maps" module establishes the fundamental geographic context on which all other data layers are overlaid. Users can switch among three types of base maps, each tailored to a specific visualization objective. The "Topographic Base Map" in Fig. 3.4(a) presents a detailed representation of both natural and built terrain features, including elevation variations, water bodies, vegetation, and major infrastructure outlines, making it suitable for analyzing landscape structure and spatial relationships. The "Imagery Base Map" in Fig. 3.4(b) consists of high-resolution satellite and aerial imagery, allowing users to observe the city from a bird's-eye perspective and enhancing visual realism through real-world textures. The "Streets Base Map" in Fig. 3.4(c) emphasizes road networks, public transportation routes, and place

names, providing a clear and schematic view of urban circulation that is particularly useful for navigation and transportation-related analysis.

The "Info Layers" section superimposes thematic geographic data on the selected base map, enabling multi-dimensional exploration of the urban environment. For clarity and usability, all information layers are organized into three domains. The Environment domain includes ecological and natural features, such as green spaces, water coverage, and environmental indicators. The Infrastructure domain represents urban systems including buildings, transportation networks, and public service facilities, with examples such as subway lines shown in Fig. 3.4(f). The People domain captures demographic and socioeconomic characteristics, incorporating layers such as industrial areas in Fig. 3.4(d) and population density in Fig. 3.4(e), which provide insight into human activity patterns and spatial disparities. These layers are presented as selectable tiles and can be interactively toggled or combined. Users may activate multiple sub-layers simultaneously to conduct comparative analysis in real time, supporting a deeper understanding of complex urban systems.

The "Sightseeing" section enables direct access to virtual tours of prominent landmarks embedded within the city model. Users can select from a curated list of attractions, each rendered with detailed three-dimensional geometry and associated contextual information. In the current implementation, this section includes Times Square in Fig. 3.4(g), the Statue of Liberty in Fig. 3.4(h), and the Brooklyn Bridge in Fig. 3.4(i). Upon selection, the system smoothly transitions the user's viewpoint to the chosen location and presents labeled visual markers and dynamic overlays. These features are designed to enhance user engagement, support educational use cases, and showcase the applicability of XR technologies in virtual tourism and urban digital twin scenarios.

By combining multiple base maps, layered thematic information, and landmark-oriented exploration, the proposed visualization framework achieves a balance between flexibility, informational richness, and usability. Supported by remote rendering and immersive XR interaction techniques, the system allows users to explore urban

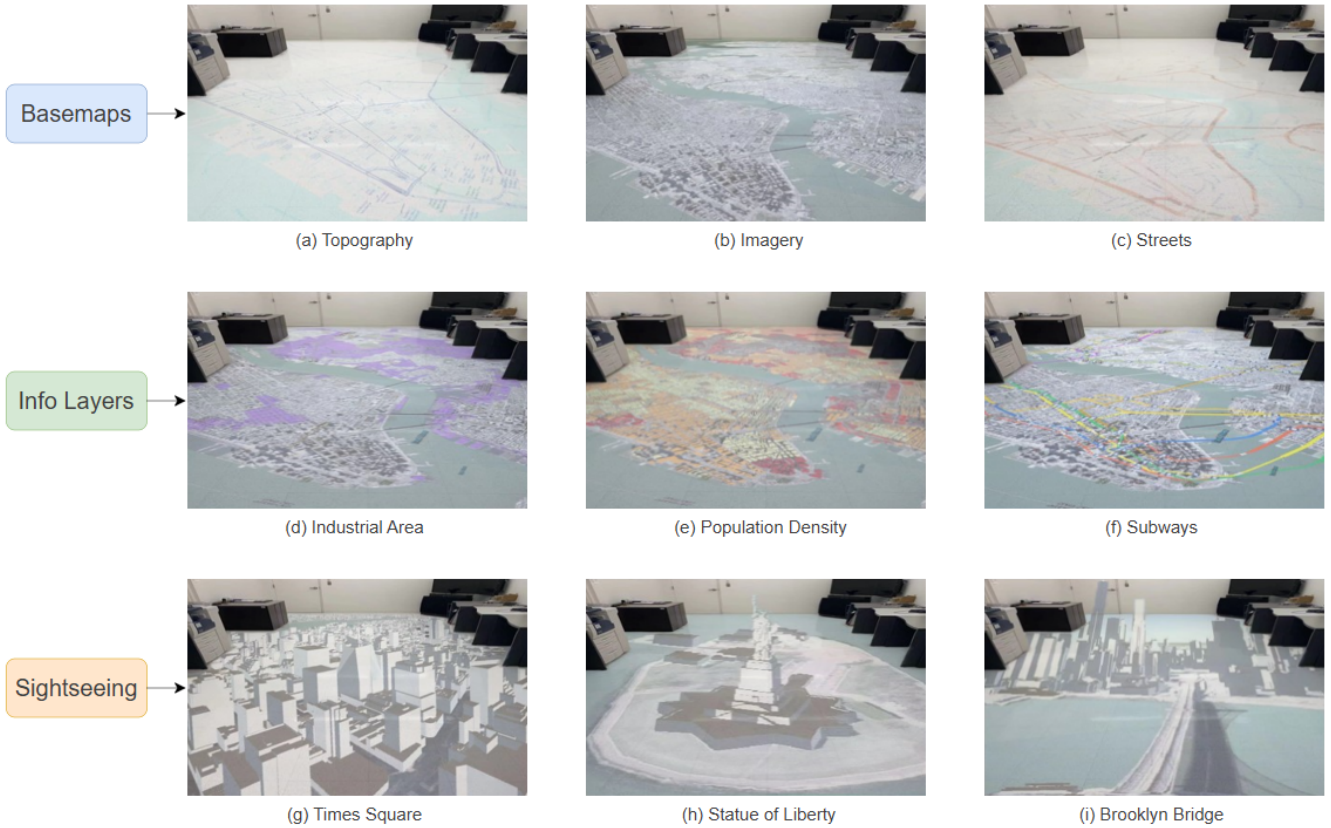


Figure 3.4: The main menu consists of three sections: "Base Maps", "Info Layers", and "Sightseeing". Representative examples are shown for each section, including topography, imagery, and streets for "Base Maps"; industrial areas, population density, and subway lines for "Info Layers"; and Times Square, the Statue of Liberty, and the Brooklyn Bridge for "Sightseeing".

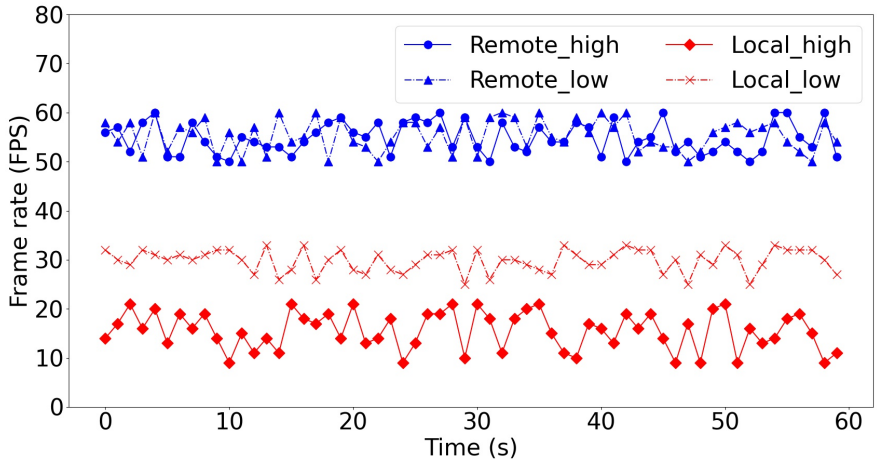
environments in a layered and semantically meaningful manner. This approach not only facilitates spatial data analysis but also enables real-time, context-aware exploration for applications such as urban planning, education, and public engagement.

3.5 Remote Rendering vs. Local Rendering

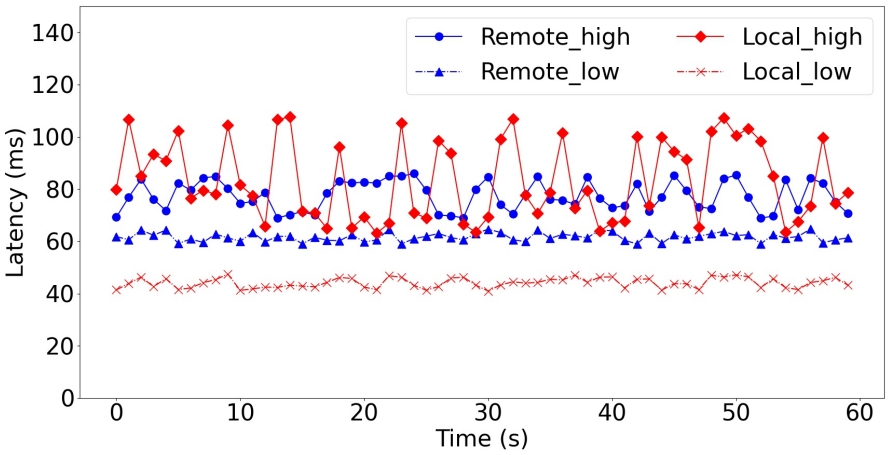
To evaluate the effectiveness of the proposed XR system based on remote rendering, a comparative performance analysis was conducted against a baseline configuration in which all rendering tasks were executed locally on the HoloLens 2. The evaluation focuses on three performance indicators that directly affect user experience in immersive applications: frame rate, rendering resolution, and end-to-end latency. Maintaining a stable frame rate and low latency is critical for ensuring visual continuity, interaction responsiveness, and overall comfort in mixed reality environments.

In both remote and local rendering configurations, the system was initially set to target a frame rate of 60 frames per second at a rendering resolution of 2048×1080 pixels. This configuration matches the native display capability of the HoloLens 2 and is widely regarded in the XR literature as a baseline requirement for smooth and immersive interaction. In addition, following established guidelines, the acceptable upper bound for end-to-end latency was set to approximately 60 milliseconds, beyond which users may experience noticeable interaction delay or motion discomfort [67].

To further examine the scalability and robustness of each rendering approach under different visual quality requirements, an additional experimental condition with reduced rendering resolution was introduced. Specifically, all measurements were repeated using a lower resolution of 1024×540 pixels while maintaining the same 60 FPS target. This low-resolution configuration represents lightweight rendering scenarios in which computational demand is reduced, allowing an assessment of whether local rendering performance can be improved through resolution scaling on resource-constrained devices such as the HoloLens 2.



(a) Frame rate



(b) Latency

Figure 3.5: Comparison between the proposed remotely rendered system and the locally rendered system in terms of frame rate and end-to-end latency.

For each resolution setting, performance was evaluated in terms of achieved frame rate, end-to-end input-to-display latency, and rendering stability. System-level resource utilization was also monitored, including CPU and GPU load on the HoloLens 2 during local rendering and on the server during remote rendering. In addition, thermal behavior and battery consumption were observed during local rendering sessions,

as sustained high computational load may result in overheating and performance throttling on mobile XR headsets. These measurements were designed to determine whether offloading rendering to an external server can provide a more stable and responsive user experience, particularly for visually complex and data-intensive scenarios.

To assess real-world performance under different rendering configurations, four representative scenarios were designed by combining rendering location and output resolution. As illustrated in Fig. 3.5, these scenarios include Remote_high (remote rendering at high resolution), Remote_low (remote rendering at low resolution), Local_high (local rendering at high resolution), and Local_low (local rendering at low resolution). Each scenario was evaluated using two key performance metrics: frame rate and end-to-end latency.

The experimental results reveal a clear advantage of remote rendering in terms of frame rate stability. Under local rendering, particularly at high resolution, the HoloLens 2 was unable to sustain acceptable performance. The measured frame rate for local high-resolution rendering ranged from 9 to 21 frames per second, indicating severe computational bottlenecks. Even when the resolution was reduced, local rendering performance remained limited, with frame rates fluctuating between 25 and 33 FPS. These values are substantially below the 60 FPS target required for smooth mixed reality interaction. In contrast, both remote rendering configurations consistently achieved frame rates close to the target of 60 FPS, independent of resolution. This demonstrates that offloading rendering tasks to a dedicated server effectively alleviates the computational constraints of the HoloLens 2 and enables real-time rendering of complex three-dimensional scenes.

Latency measurements further support these findings. The local low-resolution scenario exhibited the lowest average latency, approximately 40 milliseconds, due to the absence of network transmission overhead. However, local high-resolution rendering incurred significantly higher latency, reaching up to 110 milliseconds. Such delays can noticeably degrade interaction responsiveness and may cause user discomfort dur-

ing extended use. Although remote rendering introduces additional latency from video encoding, network transmission, and decoding, the overall delay remained within an acceptable range. Specifically, remote low-resolution rendering produced latencies between 59 and 65 milliseconds, while remote high-resolution rendering resulted in latencies between 68 and 86 milliseconds. Despite the added communication overhead, these values are substantially lower than those observed in local high-resolution rendering and remain within tolerable limits for interactive XR applications.

In summary, the comparative evaluation demonstrates that remote rendering provides clear performance advantages over local rendering on the HoloLens 2. By leveraging external computational resources, the proposed system achieves stable frame rates and controlled latency, enabling a consistent and high-quality mixed reality experience even for large-scale and visually demanding environments such as city-scale digital twins.

3.6 Metaverse Network Traffic Analysis and Modeling

Metaverse networks refer to communication infrastructures specifically designed to support the transmission of extended reality content across distributed systems [68]. In the proposed system, the metaverse network serves as the communication bridge between the HoloLens 2 headset and the remote rendering server. It is responsible for delivering user motion and interaction data from the headset to the server and streaming rendered video frames back to the user in real time.

To characterize the network traffic generated by this system, we conducted an empirical traffic analysis using Wireshark, a widely adopted packet capture and inspection tool. Traffic traces were collected directly on the rendering server during live XR sessions. The experiments were conducted under the default operating configuration of the HoloLens 2, with a target frame rate of 60 frames per second and

a rendering resolution of 2048×1080 pixels. High-Efficiency Video Coding (HEVC) was employed as the video compression standard. To avoid artificial constraints on traffic behavior, no explicit upper bound was imposed on the transmission rate. All captured packets were stored locally for offline analysis.

To decode and extract protocol-level features from the captured traces, we used Scapy, a Python-based packet manipulation and analysis library. This process enabled detailed examination of transport protocols, packet sizes, and temporal transmission patterns. The analysis indicates that the system relies on User Datagram Protocol (UDP) over IPv4 rather than Transmission Control Protocol (TCP). This design choice is consistent with the stringent latency requirements of XR systems. While TCP ensures reliable delivery through acknowledgment and retransmission mechanisms, its additional overhead and retransmission delays are ill-suited for real-time interactive applications. In contrast, UDP offers lightweight and low-latency transmission, making it more appropriate for XR scenarios in which timely delivery is prioritized over strict reliability. Limited packet loss can be tolerated as long as perceptual continuity is preserved.

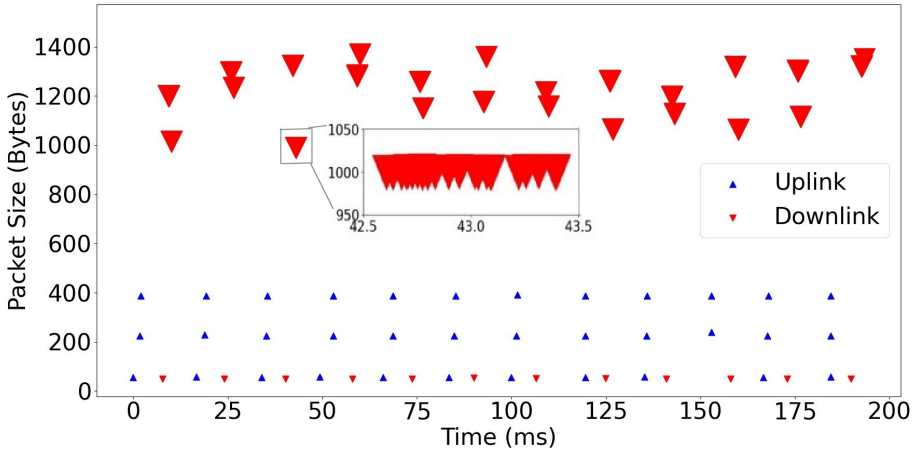


Figure 3.6: Representative packet traces from the uplink and downlink streams captured during system operation.

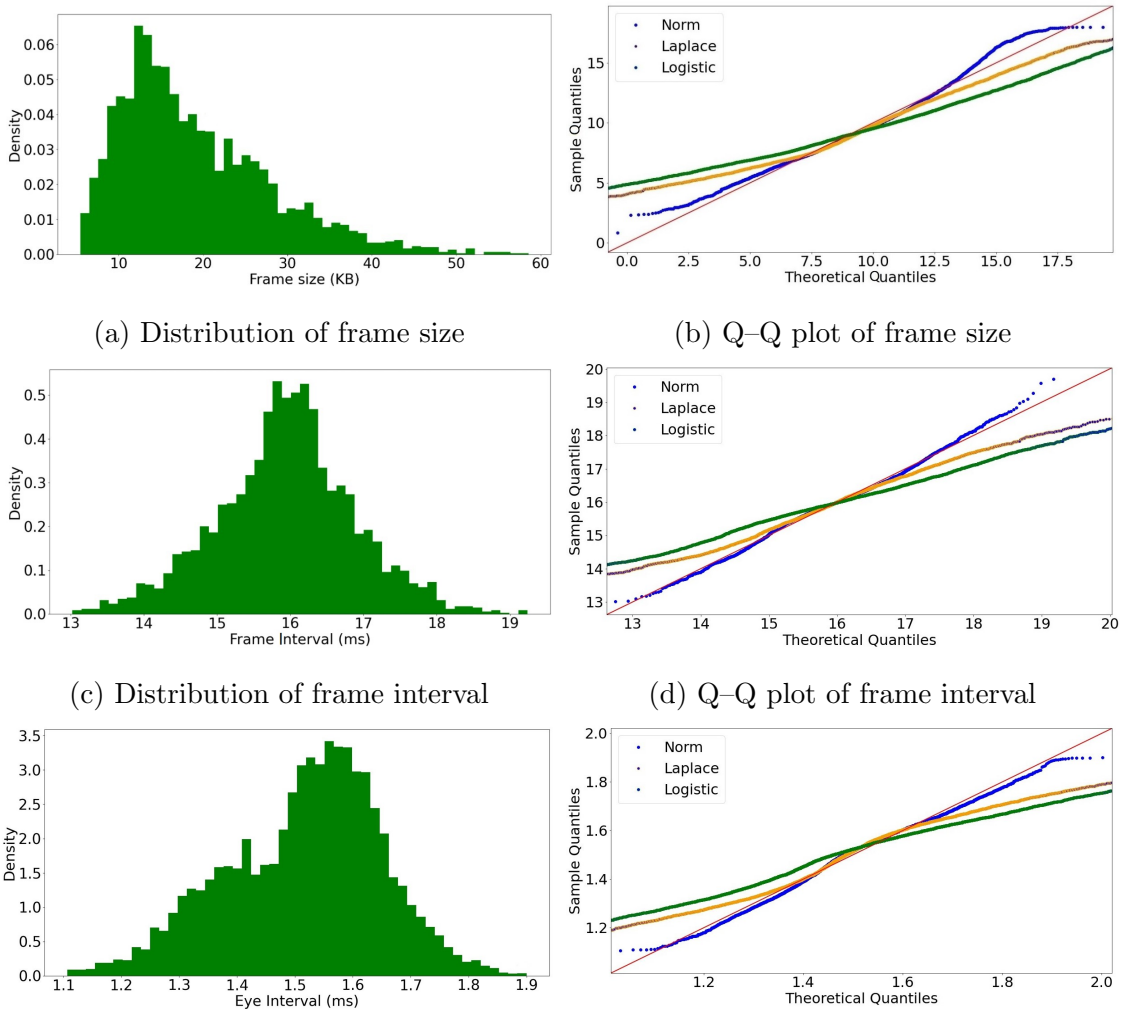
The design of high-performance XR networks requires an accurate understanding

of traffic characteristics. Precise traffic modeling not only supports optimization of real-world deployments but also enables simulation studies and the generation of synthetic metaverse traffic traces for testing and validation [68]. Among various traffic properties, two aspects are particularly relevant for system optimization: the statistical distribution of video frame data and the temporal behavior of frame transmission. To this end, we analyzed a representative segment of the bidirectional traffic stream generated during system operation. The observed uplink and downlink traffic patterns are illustrated in Fig. 3.6.

In the uplink stream, two to three small packets are transmitted from the HoloLens 2 at intervals of approximately 17 milliseconds. This interval closely matches the theoretical inter-frame duration of a 60 FPS system, which is about 16.7 milliseconds. Based on packet size and timing, these uplink packets primarily carry user interaction information, including head pose, gaze direction, and hand motion data, as well as synchronization metadata required by the rendering server.

In the downlink stream, which delivers rendered video frames and control information, two distinct packet types are observed. The first type consists of large packets, typically exceeding 1000 bytes, transmitted in short bursts composed of multiple consecutive packets. These bursts occur approximately every 17 milliseconds and often appear in closely spaced pairs. This pattern is consistent with stereo rendering, where separate video streams are generated for the left and right eyes and transmitted at the target frame rate. The second type consists of smaller packets carrying synchronization and control information. These packets are transmitted at regular intervals aligned with the frame cycle but are independent of the video data bursts.

Taken together, these observations reveal a highly regular and periodic traffic structure that is tightly coupled to the rendering pipeline of the HoloLens 2. Both uplink and downlink traffic rates are largely governed by the system frame rate, confirming that video rendering and interactive feedback dominate the traffic profile of the XR system.



(a) Distribution of frame size

(b) Q–Q plot of frame size

(c) Distribution of frame interval

(d) Q–Q plot of frame interval

(e) Distribution of eye interval

(f) Q–Q plot of eye interval

Figure 3.7: Statistical analysis of downlink traffic, including video frame size, frame interval, and eye interval. Q–Q plots compare empirical data against normal, Laplace, and logistic distributions.

Our analysis focuses on three key properties of downlink metaverse traffic: video frame size, frame arrival interval, and the inter-burst interval between left-eye and right-eye frames, referred to as the eye interval. These properties are critical for constructing realistic traffic models and for designing predictive or adaptive mechanisms in XR networks. For each property, empirical distributions were first visualized using histograms, as shown in Fig. 3.7(a), (c), and (e). Quantile-quantile plots were then

constructed to assess goodness of fit against candidate theoretical distributions. In a Q–Q plot, close alignment of sample points with the reference line $y = x$ indicates consistency between empirical and theoretical distributions [68].

We compared the empirical data with three commonly used probability distributions: normal, Laplace, and logistic. Their standardized probability density functions are given by

$$\text{Normal}(x) = \frac{e^{-x^2/2}}{\sqrt{2\pi}}, \tag{3.1}$$

$$\text{Laplace}(x) = \frac{e^{-|x|}}{2}, \tag{3.2}$$

$$\text{Logistic}(x) = \frac{e^{-x}}{(1+e^{-x})^2}. \tag{3.3}$$

As shown in Fig. 3.7(b), the Q–Q plot for frame size exhibits strong alignment with the normal distribution, while noticeable deviations appear for the Laplace and logistic distributions. Similar trends are observed for the frame interval in Fig. 3.7(d) and for the eye interval in Fig. 3.7(f). In all cases, the normal distribution provides the closest fit to the empirical data.

Based on these results, we conclude that video frame size, frame interval, and eye interval in the studied XR system can be well approximated by normal distributions. This finding provides a practical statistical foundation for modeling metaverse traffic and supports the development of simulation frameworks and performance evaluation tools in future work.

Accurate prediction of video frame sizes in XR systems remains challenging due to the temporal complexity of metaverse traffic. Unlike conventional video streaming, XR traffic exhibits both long-range dependence and short-range dependence, resulting in persistent correlations over extended time horizons as well as rapid short-term fluctuations. These characteristics complicate the construction of predictive models that are both stable and responsive. While existing studies have proposed traffic prediction methods for HTTP-based adaptive streaming protocols such as Dynamic Adaptive Streaming (DAS) over HTTP and HTTP Live Streaming (HLS) [69], those

approaches are tailored to segment-based, request-response architectures with significant buffering and are not directly applicable to XR systems.

XR applications typically rely on Real-time Transport Protocol for continuous, low-latency media delivery. In addition, metaverse traffic exhibits strong asymmetry between downlink and uplink streams. The downlink is dominated by high-throughput video data, often referred to as an elephant flow, whereas the uplink consists of frequent but lightweight packets carrying user interaction data, forming a mice flow. This dual-flow structure differs fundamentally from traditional video streaming traffic and motivates the need for XR-specific prediction models.

Given the limited prior work on metaverse traffic prediction, we propose a statistical approach to forecast successive video frame sizes. Let F_t denote the frame size at time t, forming a univariate time series. We model this series using an Autoregressive Moving Average model of order (p, q), defined as

$$F_t = c + \epsilon_t + \sum_{i=1}^{p} \phi_i F_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}, \tag{3.4}$$

where ϕ_i and θ_i are the autoregressive and moving average coefficients, respectively, ϵ_t denotes white noise, and c is a constant term.

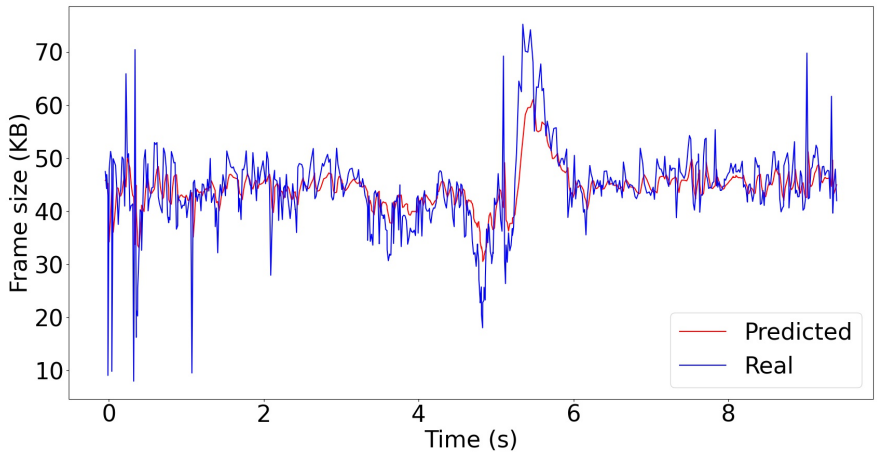


Figure 3.8: Predicted frame sizes obtained from the ARMA-based model compared with ground truth frame sizes on the test dataset.

Stationarity is a prerequisite for Autoregressive Moving Average (ARMA) modeling. To verify this condition, we applied the Augmented Dickey–Fuller test to the frame size series, which confirmed stationarity at the 95% confidence level. Model orders were selected by examining the Autocorrelation Function and Partial Autocorrelation Function, leading to the choice of $p = 5$ and $q = 4$.

For evaluation, 70% of the dataset was used for training and the remaining 30% for testing. As illustrated in Fig. 3.8, the predicted frame sizes closely track the ground truth values, including during periods of pronounced fluctuation. These results demonstrate that the ARMA(5,4) model effectively captures the temporal dependencies in metaverse frame size dynamics, making it a suitable candidate for short-term traffic prediction in future adaptive streaming and bandwidth estimation mechanisms.

3.7 Limitations

Although this chapter demonstrates the feasibility of remote rendering for supporting complex metaverse applications on resource-constrained XR devices, several limitations should be acknowledged.

First, the experimental evaluation is based on a specific prototype system and a single representative case study. In this chapter, the proposed system is implemented using Microsoft HoloLens 2, Unity, Holographic Remoting, and a New York City digital twin dataset. This setup provides a realistic and visually demanding environment for evaluating the potential of remote rendering. However, the findings may not be directly generalizable to all XR devices, rendering engines, or metaverse applications. Different head-mounted displays may have different display resolutions, tracking mechanisms, decoding capabilities, thermal constraints, and onboard computing resources. Similarly, different metaverse scenarios, such as indoor industrial training, multi-user social interaction, collaborative design, or highly dynamic virtual gaming environments, may introduce different rendering workloads and interaction

patterns. Therefore, while the case study provides useful evidence for the advantages of remote rendering, further validation across more heterogeneous XR devices, application domains, and scene complexities is needed to strengthen the generality of the conclusions.

Second, the system is evaluated under relatively controlled network and deployment conditions. In the current implementation, the HoloLens 2 is connected to the rendering server through a USB-C cable to ensure stable data transmission and repeatable performance measurement. This design is useful for isolating the impact of rendering offloading from uncontrolled wireless network fluctuations. However, practical metaverse systems are more likely to operate over Wi-Fi, 5G, or edge-cloud networks, where bandwidth variation, packet loss, jitter, handover delay, and network congestion may significantly affect the user experience. In addition, this chapter mainly considers a single-user remote rendering scenario, whereas real-world metaverse platforms often involve multiple users sharing the same edge server and network infrastructure. As the number of users increases, server-side GPU contention, bandwidth competition, and scheduling conflicts may become important performance bottlenecks. Therefore, future work should evaluate the proposed approach under realistic wireless networks and multi-user edge rendering environments.

Third, the evaluation mainly relies on objective system-level measurements and trace-based traffic modeling, while subjective user experience and broader traffic dynamics are not fully explored. The comparison between remote and local rendering focuses on metrics such as frame rate, resolution, latency, and resource utilization, which are essential for assessing XR system performance. However, these metrics cannot fully represent user-perceived QoE, including visual comfort, interaction naturalness, perceived responsiveness, cybersickness, and overall immersion. A larger-scale user study would be necessary to examine whether the measured performance improvements translate into meaningful perceptual benefits. Moreover, the network traffic analysis is based on traces collected from one specific system configuration, including HoloLens 2, Holographic Remoting, HEVC encoding, 60 FPS, and 2048×1080

resolution. Although the results reveal useful traffic characteristics and show that ARMA-based modeling can capture short-term frame size dynamics, the model may not fully capture more complex traffic behaviors caused by abrupt scene changes, diverse user interactions, adaptive bitrate control, or different codec configurations. Future research should combine subjective QoE studies with more diverse traffic traces and advanced prediction models to provide a more comprehensive understanding of metaverse network behavior.

Chapter 4

A Collaborative Thing-Edge-Cloud Architecture for the Metaverse

4.1 Core Components

The proposed agentic metaverse architecture is a hierarchical thing–edge–cloud system (Fig. 4.1), in which heterogeneous agents are deployed across layers with clearly differentiated responsibilities. This architectural organization is motivated by the intrinsic heterogeneity of metaverse workloads, which simultaneously involve high-frequency user interaction, latency-sensitive control decisions, and computation-intensive cognitive reasoning. By decoupling perception, execution, and cognition across layers while maintaining coordinated control, the framework supports both scalability and responsiveness in complex immersive environments. Moreover, the layered design provides a principled foundation for integrating agentic intelligence into metaverse systems in a systematic and extensible manner. The core components of the architecture are described below.

- Thing layer: The thing layer represents the embodied interaction endpoint of the agentic metaverse, where users directly engage with virtual environments through immersive devices such as XR headsets, controllers, and sensing units. This layer continuously captures fine-grained user inputs, including body motion, hand gestures, head orientation, gaze direction, and interaction commands,

and delivers multimodal feedback encompassing visual, auditory, and haptic signals under strict latency constraints. As the closest layer to human perception and action, the thing layer emphasizes responsiveness, stability, and perceptual fidelity rather than complex computation. By anchoring system intelligence in real-time physical interaction, the thing layer establishes the perceptual grounding of the architecture and serves as both the input and output interface of the perception–decision–action cycle.

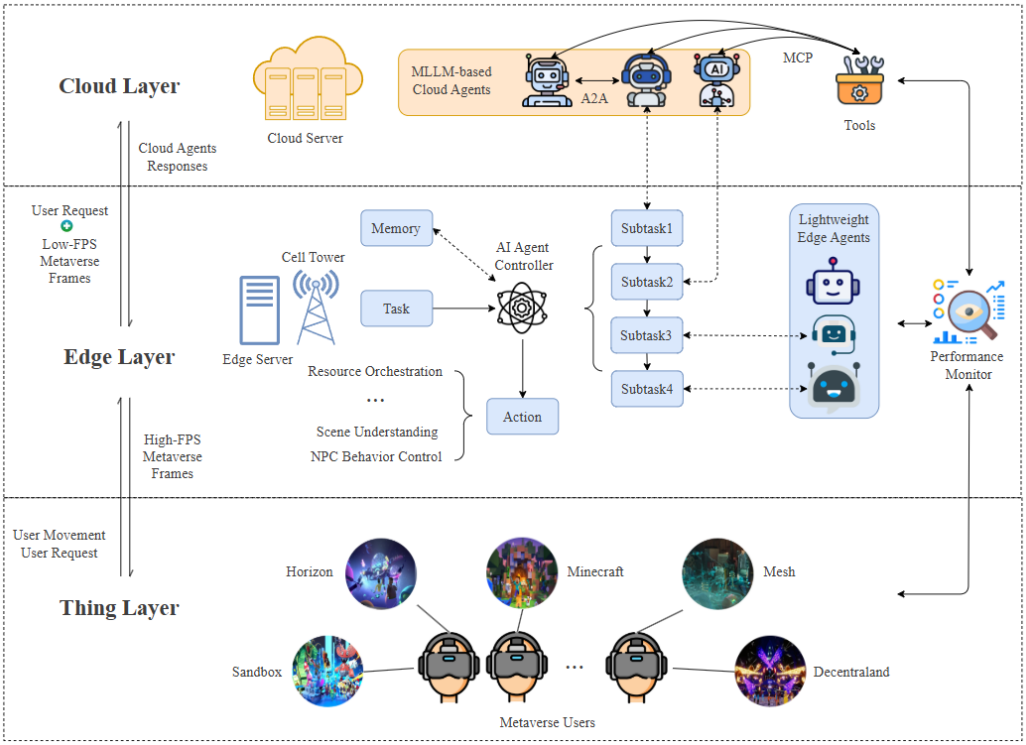


Figure 4.1: Proposed agentic metaverse architecture based on a thing–edge–cloud hierarchy. The framework integrates MLLM-based cognitive agents at the cloud layer, lightweight agents for real-time execution at the edge layer, and immersive user interaction at the thing layer. An AI agent controller coordinates task decomposition, resource allocation, and agent collaboration, while a performance monitoring mechanism supports adaptive and closed-loop system operation.

- Edge layer with lightweight edge agents: The edge layer provides real-time execution capabilities and is deployed on geographically distributed edge servers

in close proximity to end users. Its primary role is to ensure low-latency responsiveness and local adaptability under fluctuating network conditions and dynamic user interactions. Lightweight edge agents residing in this layer are responsible for time-sensitive operations such as adaptive bandwidth allocation, rendering control, and short-horizon decision-making. These agents typically rely on compact models or reinforcement learning policies that can be executed efficiently with limited computational overhead. By prioritizing fast reaction and local optimization, the edge layer enables stable and continuous interaction while reducing end-to-end latency and mitigating bandwidth pressure on the cloud layer.

- Cloud layer with MLLM-based cloud agents: The cloud layer functions as the centralized computation and information backbone of the agentic metaverse. Supported by abundant computing and storage resources, this layer enables persistent data management, global state maintenance, and large-scale model inference. MLLM-based cloud agents operate within this layer to conduct computation-intensive cognitive tasks, including multimodal reasoning, long-context understanding, knowledge integration, and semantic abstraction. By aggregating visual, textual, and interaction data across users and time, these agents construct high-level representations of the virtual environment and user intent. Such representations provide the cognitive basis for long-term consistency, global coordination, and strategic decision-making across the metaverse system.

- AI agent controller: The AI agent controller serves as the orchestration core that coordinates heterogeneous agents across the thing–edge–cloud hierarchy to achieve coherent and goal-oriented behavior. It maintains a unified system perspective by continuously tracking user interactions, environmental context, execution states, and performance feedback from all layers. Based on this global awareness, the controller performs task cognition and planning by decomposing high-level objectives into structured subtasks with explicit dependencies and

temporal constraints. For each subtask, the controller dynamically selects the execution layer by jointly considering latency sensitivity, computational complexity, context length requirements, data locality, and current system load. Continuous runtime monitoring further enables adaptive scheduling, load balancing, and re-planning, thereby improving system robustness under dynamic and uncertain operating conditions.

Overall, the proposed framework operates through a hierarchical and iterative processing pipeline that integrates embodied interaction, real-time execution, and cloud-scale cognition. The pipeline begins at the thing layer, where continuous user interactions define the system context through embodied inputs and feedback signals. These signals are partially processed at the edge layer to support low-latency adaptation and responsive execution, while distilled contextual information and performance summaries are forwarded to the cloud layer. At the cloud layer, MLLM-based agents perform computation-intensive reasoning over aggregated and historical context to derive high-level semantic representations and long-term strategies. Guided by these representations, the agent controller organizes tasks, decomposes objectives, and orchestrates coordinated execution across layers. Execution outcomes and performance indicators are continuously fed back into the control loop, enabling adaptive refinement and closed-loop coordination. Through this integrated pipeline, the architecture unifies perception, reasoning, and action into a coherent agentic intelligence process tailored for scalable and immersive metaverse systems.

4.2 Key Performance Metrics

Evaluating the proposed agentic metaverse architecture requires a set of performance metrics that go beyond conventional network-centric or device-centric indicators. Unlike traditional metaverse systems that primarily focus on rendering quality or communication efficiency, the proposed framework tightly integrates embodied interaction

at the thing layer, real-time adaptive execution at the edge layer, and cloud-scale cognitive reasoning coordinated by a centralized AI agent controller. As a result, system effectiveness must be assessed from multiple complementary dimensions that jointly capture intelligence, efficiency, robustness, and user-perceived experience. To this end, the following categories of performance metrics are defined to provide a unified and comprehensive evaluation perspective.

- Agent cognition metrics: These metrics assess the ability of intelligent agents to perceive environmental context, interpret multimodal inputs, infer user intent, and generate coherent decision outcomes over time. They characterize the quality, consistency, and temporal stability of semantic understanding across successive decision cycles, as well as the alignment between inferred goals, planned actions, and observed system behavior. Within the proposed architecture, agent cognition metrics primarily reflect the reasoning effectiveness of MLLM-based cloud agents and the fidelity with which their cognitive outputs are translated into executable strategies by the AI agent controller.

- Resource utilization metrics: These metrics evaluate how efficiently computational, rendering, and networking resources are allocated and utilized across the cloud and edge layers. Rather than focusing solely on absolute resource consumption, they emphasize utilization effectiveness relative to achieved performance and service quality. Such metrics capture the adaptability of agent-driven resource control under dynamic workloads, the stability of allocation decisions over time, and the coordination overhead introduced by cross-layer orchestration. They provide insight into how well the system balances performance gains against operational costs.

- Task achievement metrics: Task achievement metrics measure the extent to which user-specified objectives or system-level goals are successfully realized. They assess task completion accuracy, execution timeliness, and responsiveness to changes in user intent or environmental conditions. In addition, these met-

rics evaluate the consistency between intended outcomes derived during task planning and the actual execution results observed at runtime. By directly linking agent decision-making processes to functional outcomes, task achievement metrics reflect the practical effectiveness of agentic control in supporting goal-oriented interaction.

- Immersive experience metrics: These metrics evaluate the perceptual quality, continuity, and stability of user experience within the metaverse environment. They capture aspects such as smoothness of interaction, visual and temporal coherence, and resilience of immersion under fluctuating system load or network conditions. By prioritizing user-perceived experience rather than isolated low-level indicators, immersive experience metrics assess the tangible impact of agentic coordination, adaptive execution, and resource management on overall immersion quality.

- Robustness and adaptability metrics: Robustness and adaptability metrics characterize the system's ability to maintain stable and reliable operation under unexpected or adverse conditions, including abrupt changes in user interaction patterns, variable resource availability, network disturbances, and partial agent failures. These metrics evaluate the effectiveness of self-adaptive mechanisms, fault tolerance strategies, and recovery processes enabled by the agentic framework. They reflect the capacity of the architecture to sustain acceptable performance and user experience in dynamic and uncertain operating environments.

Together, these metric categories form a holistic evaluation framework that captures cognitive intelligence, operational efficiency, experiential quality, and system resilience. By jointly considering these dimensions, the proposed metrics enable a systematic assessment of how well the agentic metaverse architecture integrates perception, reasoning, and action across the thing–edge–cloud hierarchy.

4.3 Communication Protocols

Efficient coordination across heterogeneous layers and intelligent agents requires a communication design that is closely aligned with functional roles and interaction patterns within the system. The proposed framework therefore adopts a role-aware and differentiated communication strategy, in which protocol selection is guided by the latency requirements, semantic complexity, and coordination objectives of each communication path. Instead of relying on a monolithic protocol stack, the architecture integrates multiple complementary communication mechanisms, allowing each layer and agent interaction to operate under protocols that best match its operational characteristics.

- Thing–edge communication: The communication between the thing layer and the edge layer underpins embodied interaction and immersive user experience, and thus emphasizes ultra-low latency, continuous bidirectional transmission, and resilience to rapidly changing network conditions. This communication path carries high-frequency interaction events and real-time media streams, where timely delivery is more critical than strict reliability guarantees. Protocols such as WebRTC naturally fit this interaction pattern by providing native support for real-time audio and video transmission together with adaptive congestion control [70]. Transport solutions built on QUIC further reduce connection setup overhead and improve responsiveness under dynamic network conditions, while lightweight persistent channels such as WebSocket complement media transport by enabling interaction control, signaling, and state synchronization [71]. Collectively, these mechanisms enable smooth and responsive thing–edge interaction without introducing excessive protocol overhead.

- Edge–cloud communication: Communication between the edge and cloud layers focuses on exchanging abstract system context, task specifications, execution feedback, and resource coordination information. In contrast to thing–edge

interaction, this communication path is less constrained by hard real-time requirements and instead prioritizes reliability, scalability, and semantic correctness. Service-oriented communication based on RESTful APIs over HTTP/2 or HTTP/3 provides standardized and interoperable interfaces for accessing cloud services, while gRPC enables efficient structured communication suitable for coordination tasks that remain performance sensitive [72]. For large-scale and distributed deployments, message-oriented middleware such as Kafka or RabbitMQ supports asynchronous communication and decoupling between cloud-level cognition and edge-level execution, improving system robustness and scalability [73, 74].

- Agent–agent communication: Communication among intelligent agents distributed across the cloud and edge layers is primarily semantic and goal-driven. Rather than transmitting raw sensory data, these interactions focus on sharing intent, intermediate reasoning outcomes, coordination signals, and task-related state. Agent-to-Agent communication paradigms provide an abstraction well suited to collaborative planning, negotiation, and distributed decision-making [75]. Contemporary multi-agent frameworks adopt similar mechanisms, including structured conversational exchanges, stateful message-passing graphs, and lightweight peer-to-peer coordination models, enabling agents to align their behaviors while preserving autonomy.

- Agent–tool communication: Agent–tool communication enables intelligent agents to access external computational services, models, or knowledge bases as part of their reasoning and execution workflow. This interaction is inherently task-oriented and requires clearly defined interfaces with predictable execution semantics. The Model Context Protocol offers an open and standardized approach for exposing tool capabilities to agents in a unified and interpretable format [76]. In parallel, function-calling mechanisms and OpenAPI-based interfaces provide widely adopted solutions for structured service invocation, supporting modular integration, extensibility, and interoperability within agent-centric systems.

By adopting a differentiated communication design, the proposed framework aligns protocol choices with the functional requirements of each interaction path. This approach enables low-latency embodied interaction, reliable and scalable coordination between edge and cloud, and semantically rich collaboration among agents and tools, thereby supporting coherent and efficient operation in large-scale agentic metaverse environments.

4.4 Limitations

Although this chapter proposes a collaborative thing–edge–cloud architecture for supporting agentic intelligence in the metaverse, several limitations should be acknowledged.

First, the proposed architecture is mainly presented as a conceptual and system-level framework, rather than a fully deployed large-scale metaverse system. The framework defines the functional roles of the thing layer, edge layer, cloud layer, and AI agent controller, and explains how heterogeneous agents can collaborate across these layers. However, the complete architecture has not yet been implemented and evaluated as an end-to-end production system. In particular, the deployment of MLLM-based cloud agents, lightweight edge agents, and the AI agent controller across real edge-cloud infrastructure may introduce additional engineering challenges, including model serving latency, cross-layer synchronization, runtime scheduling overhead, and resource contention. Therefore, while the proposed architecture provides a structured foundation for integrating agentic intelligence into metaverse systems, further implementation and experimental validation are needed to examine its practical feasibility, scalability, and efficiency under real-world deployment conditions.

Second, the proposed performance metrics provide a holistic evaluation perspective, but they remain relatively high-level and require further operationalization. This chapter identifies several important metric categories, including agent cognition, resource utilization, task achievement, immersive experience, and robustness. These

dimensions are useful for understanding the multi-faceted nature of agentic metaverse systems. However, some of these metrics, especially agent cognition and task achievement, are difficult to quantify in a unified and objective manner. For example, evaluating whether an agent correctly understands user intent, maintains semantic consistency over time, or successfully decomposes a complex task may depend on task context, user expectations, and evaluation criteria. Similarly, immersive experience metrics often require subjective user studies in addition to system-level measurements. Future work should define more concrete measurement methods, benchmark tasks, and evaluation protocols to transform these metric categories into reproducible quantitative indicators.

Third, the communication protocol design is discussed from a functional perspective, but its performance, interoperability, and security implications are not fully evaluated in this chapter. The proposed architecture suggests different communication mechanisms for thing–edge, edge–cloud, agent–agent, and agent–tool interactions, such as WebRTC, QUIC, RESTful APIs, gRPC, message-oriented middleware, and agent communication protocols. However, the integration of multiple protocols in a unified metaverse system may introduce additional complexity. For instance, different communication paths may have different latency, reliability, serialization, synchronization, and fault-tolerance requirements. In addition, agent–agent and agent–tool communication may involve sensitive user context, multimodal data, and high-level reasoning outputs, raising concerns about privacy protection, access control, and secure coordination. Therefore, future research should further investigate protocol-level performance, cross-platform interoperability, and security mechanisms to ensure that the proposed architecture can operate reliably in open, dynamic, and large-scale metaverse environments.

Chapter 5

Deep Reinforcement Learning for Resource Allocation in the Metaverse

5.1 Problem Formulation

In a edge computing-enabled metaverse, a large number of users simultaneously access immersive services with strict requirements on latency, throughput, and computational capacity. Representative services include real-time rendering of virtual environments, AI-driven interaction, and personalized content streaming. These services jointly impose heavy and highly dynamic loads on both communication and computation resources at the network edge. When resource management is inefficient, congestion and contention quickly emerge, leading to degraded QoE and limited system scalability. As a result, efficient and adaptive resource allocation is a fundamental prerequisite for sustaining high-quality metaverse services.

We formulate the resource allocation problem as a cooperative multi-agent decision-making process, in which multiple agents collaboratively allocate constrained edge resources (e.g., wireless bandwidth, CPU cycles, and GPU memory) to users or tasks. Each agent determines its local allocation decision based on partial observations of the system state. This formulation naturally captures the decentralized, dynamic, and interactive characteristics of metaverse systems, where user demands evolve over time, network conditions fluctuate, and edge resources are inherently limited.

In this setting, we consider a fully cooperative multi-agent paradigm [77], where all agents jointly optimize a shared global objective. Specifically, the goal is to maximize overall QoE while maintaining fairness and balanced service quality across users. Consequently, agents are not optimized for individual utilities, but instead learn coordinated policies that contribute to a common long-term system objective.

Due to the distributed nature of the system, each agent has access only to local and partial observations of the global environment. For example, an agent may observe the channel condition, rendering workload, or latency requirement of a specific user, while the full system state remains unobservable. To explicitly model this partial observability and decentralized control structure, we formulate the problem as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [78].

A Dec-POMDP is defined by the tuple $(\mathcal{S}, \mathcal{N}, \{\mathcal{A}^i\}_{i\in\mathcal{N}}, P, R, \{\mathcal{O}^i\}_{i\in\mathcal{N}}, \Omega, \gamma)$, where:

- $\mathcal{S}$ denotes the global state space, which captures system-level information such as user demand, network load, edge server capacity, and service requirements.
- $\mathcal{N} = \{1, 2, \ldots, N\}$ is the set of agents, each corresponding to a user or an edge resource allocator.
- $\mathcal{A}^i$ is the action space of agent i. At each time step t, agent i selects an action $a_t^i \in \mathcal{A}^i$ according to its policy $\pi^i(a_t^i \mid o_t^i)$, and the joint action is denoted as $\mathbf{a}_t = (a_t^1, \ldots, a_t^N)$.
- $\mathcal{O}^i$ represents the observation space of agent i. Each agent receives a local observation $o_t^i \in \mathcal{O}^i$, which is generated from the global state via the observation function $\Omega(o_t^i \mid s_t)$.
- $P(s_{t+1} \mid s_t, \mathbf{a}_t)$ defines the state transition dynamics, which are governed by user behavior, service requests, and edge-side resource responses.
- $R(s_t, \mathbf{a}_t)$ is a shared reward function received by all agents, which measures system-level performance in terms of QoE and resource efficiency.

- $\gamma \in (0, 1]$ is the discount factor that balances immediate performance and long-term system utility.

This Dec-POMDP formulation provides a rigorous mathematical foundation for applying cooperative multi-agent reinforcement learning to metaverse resource allocation. In the following sections, we further detail the design of the state representation, local observation space, action space, and reward function, which together enable coordinated decision-making under partial observability.

5.1.1 State and Observation Space

In the edge computing-enabled metaverse environment, each agent corresponds to an individual user or service task, and its decision-making capability critically depends on the fidelity of its environmental perception. Owing to the decentralized control structure, agents do not have access to the full global system state. Instead, each agent relies on a local observation that reflects both communication conditions and edge-side computational performance, which jointly determine the perceived QoE and instantaneous resource demand.

At time step t, the network condition observed by agent i is denoted as N_t^i and represented as a six-dimensional vector

$$N_t^i = (x_t^i, y_t^i, l_t^i, j_t^i, p_t^i, n_t^i), \tag{5.1}$$

where x_t^i denotes the previously selected target bit rate, and y_t^i represents the actual received bit rate measured at the user side. The term l_t^i captures the average round-trip latency, reflecting end-to-end communication delay, while j_t^i characterizes network jitter, which is closely related to temporal stability of visual quality. The variables p_t^i and n_t^i denote the number of lost packets and the number of negative acknowledgment messages, respectively, both of which indicate transmission reliability and decoding robustness.

In addition to communication-related information, each agent also observes metrics associated with its rendering task at the edge server. These metrics are summarized by a four-dimensional vector

$$C_t^i = (z_t^i, u_t^i, e_t^i, d_t^i), \tag{5.2}$$

where z_t^i represents the most recent control signal issued by the agent to regulate CPU usage, and u_t^i denotes the proportion of currently available CPU resources at the edge server. The rendered frame rate for the user is captured by e_t^i, measured in frames per second, while d_t^i denotes the average rendering delay per frame, which directly affects visual smoothness and interaction responsiveness.

The complete local observation of agent i at time t is defined as the concatenation of network and computation features

$$\begin{aligned} o_t^i &= (N_t^i, C_t^i) \\ &= (x_t^i, y_t^i, l_t^i, j_t^i, p_t^i, n_t^i, z_t^i, u_t^i, e_t^i, d_t^i). \end{aligned} \tag{5.3}$$

This observation design enables each agent to jointly reason about end-to-end communication quality and edge-side rendering efficiency, thereby supporting adaptive and QoE-aware resource allocation decisions. While execution is fully decentralized, effective coordination among agents during learning requires access to global information. Accordingly, the global state $\boldsymbol{s}_t$ is defined as the aggregation of local observations from all agents

$$\boldsymbol{s}_t = (o_t^1, o_t^2, \ldots, o_t^N) \in \mathcal{S}, \tag{5.4}$$

where $\mathcal{S}$ denotes the global state space. This centralized state representation is employed exclusively during training to facilitate coordination and credit assignment, whereas decentralized execution is preserved during inference. Such a formulation is consistent with the centralized training and decentralized execution paradigm and provides a principled foundation for cooperative multi-agent learning under partial observability.

5.1.2 Action Space

In the proposed multi-agent resource allocation framework, each agent dynamically controls both communication and computation resources allocated to its associated user. Consequently, the action space must support continuous decision-making over two tightly coupled dimensions, namely transmission bit rate and CPU resource allocation for edge rendering.

For communication control, agent i selects a target transmission bit rate for rendered frames. This decision directly affects visual fidelity as well as network congestion. Let $a_t^{i,m}$ denote the communication-related action of agent i at time t, which is constrained to the interval

$$a_t^{i,m} \in [b_{\min}, b_{\max}], \tag{5.5}$$

where $b_{\min}$ and $b_{\max}$ denote the minimum and maximum allowable bit rates, respectively. These bounds are determined by hardware limitations and application-level quality-of-service requirements.

Simultaneously, the agent adjusts computation resources through CPU throttling at the edge server. Let $a_t^{i,p}$ denote the computation-related action, representing the proportion of CPU resources allocated to the rendering task. This variable is bounded as

$$a_t^{i,p} \in [l_{\min}, l_{\max}], \tag{5.6}$$

where $l_{\min}$ and $l_{\max}$ correspond to the minimum and maximum CPU usage levels expressed as fractions of total available edge CPU capacity. By tuning this parameter, the agent can balance rendering performance against overall resource efficiency.

The complete action of agent i at time t is therefore defined as a two-dimensional continuous vector

$$a_t^i = (a_t^{i,m}, a_t^{i,p}), \tag{5.7}$$

which jointly determines the communication quality and computational responsiveness experienced by the user. The individual action space of each agent is given

by

$$\mathcal{A}^i \subseteq [b_{\min}, b_{\max}] \times [l_{\min}, l_{\max}] \subset \mathbb{R}_+^2. \tag{5.8}$$

The joint action of all agents at time t is defined as

$$\boldsymbol{a}_t = (a_t^1, a_t^2, \ldots, a_t^N) \in \mathcal{A}, \tag{5.9}$$

where $\mathcal{A} = \mathcal{A}^1 \times \mathcal{A}^2 \times \cdots \times \mathcal{A}^N$ denotes the global action space. This joint action drives the environment transition and captures the collective impact of distributed resource allocation decisions. The continuous nature of the action space makes it particularly suitable for actor-critic based reinforcement learning methods and enables fine-grained adaptation to the highly dynamic and non-linear conditions characteristic of metaverse systems.

5.1.3 Reward Design

In our experiments, agents are guided by a shared reward signal that reflects collective system performance and encourages coordinated behavior. In the edge computing-enabled metaverse considered in this work, multiple agents jointly allocate communication and computation resources under dynamic and uncertain conditions. The reward function therefore plays a central role in aligning decentralized actions with global optimization objectives. To capture the multi-dimensional nature of immersive service delivery, the reward is designed to incorporate three complementary components, namely user-perceived Quality of Experience, balance in communication resource allocation, and balance in computation resource allocation. This subsection focuses on the modeling of user-perceived QoE, while subsequent components are introduced to explicitly enforce fairness and system-level efficiency.

We adopt a time-step-based QoE formulation that jointly accounts for scene quality, rendering smoothness, network latency, and temporal stability. For agent i at time step t, the instantaneous QoE is defined as

$$QoE_t^i = \alpha\, q(y_t^i) - \beta\, |f_t^i - f_{\text{target}}^i| - \gamma\, p(l_t^i) - \delta\, |q(y_{t+1}^i) - q(y_t^i)|, \tag{5.10}$$

where each term captures a distinct aspect of perceptual experience.

The first term $q(y_t^i)$ represents scene quality as a function of the received bit rate y_t^i. To model diminishing perceptual returns at high bit rates, a logarithmic utility function is employed, given by $q(y_t^i) = \log(y_t^i / y_{\min}^i)$, where $y_{\min}^i$ denotes a minimum reference bit rate. The weighting coefficient α controls the relative importance of visual fidelity.

The second term penalizes temporal choppiness caused by frame loss or delayed rendering. Here, f_t^i denotes the actual received frame rate, and f_{target}^i is the desired target frame rate. The coefficient β adjusts the sensitivity of the QoE to deviations from smooth playback.

The third term models the negative impact of network latency on user comfort and interaction responsiveness. The penalty function is defined as $p(l_t^i) = \exp(l_t^i / l_{\min}^i)$, where l_t^i denotes the observed round-trip latency and $l_{\min}^i$ serves as a normalization baseline. This exponential form reflects the empirical observation that user tolerance to latency degrades rapidly once a critical threshold is exceeded.

The final term penalizes temporal instability in scene quality across consecutive time steps. Large fluctuations in bit rate may cause visual discomfort and reduce immersion, even when the average quality remains high. The coefficient δ regulates the strength of this smoothness constraint.

The system-level QoE at time step t is obtained by aggregating individual experiences across all agents,

$$Q_t = \sum_{i=1}^{N} QoE_t^i, \tag{5.11}$$

which serves as the primary performance indicator for immersive service quality.

While maximizing aggregate QoE is essential, unconstrained optimization may result in severe resource imbalance across users. Such imbalance can degrade fairness, reduce infrastructure efficiency, and lead to unstable service behavior. To explicitly discourage disproportionate allocation of resources, we introduce additional

penalty terms that capture the dispersion of communication and computation resources among agents.

The balance of communication resources is measured through the variance of received bit rates across agents. Specifically, the communication imbalance at time step t is defined as

$$V_t^{\text{comm}} = \frac{1}{N-1} \sum_{i=1}^{N} \left(y_t^i - \frac{1}{N} \sum_{j=1}^{N} y_t^j \right)^2, \tag{5.12}$$

where a smaller value indicates a more even distribution of bandwidth.

Similarly, computation balance is evaluated based on the variance of CPU utilization across agents. Let u_t^i denote the proportion of CPU resources allocated to agent i at time t. The computation imbalance is given by

$$V_t^{\text{comp}} = \frac{1}{N-1} \sum_{i=1}^{N} \left(u_t^i - \frac{1}{N} \sum_{j=1}^{N} u_t^j \right)^2. \tag{5.13}$$

This term penalizes excessive concentration of computation resources on a subset of users and promotes load balancing at the edge server.

Combining the above components, the overall reward at time step t is defined as

$$r_t = w_1 Q_t - w_2 V_t^{\text{comm}} - w_3 V_t^{\text{comp}}, \tag{5.14}$$

where w_1, w_2, and w_3 are non-negative weighting coefficients that balance perceptual quality against fairness and efficiency considerations. The variance terms are subtracted to ensure that higher imbalance leads to lower reward.

Given the state space, action space, and reward function defined above, the objective of the cooperative multi-agent system is to learn an optimal joint policy π^* that maximizes the expected cumulative return. Formally, the optimization problem is expressed as

$$\pi^* = \arg\max_{\pi} \ \mathbb{E}_{\pi} \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \right], \tag{5.15}$$

where $\gamma \in (0, 1]$ is the discount factor.

To facilitate policy optimization, we define the state value function

$$V^{\pi}(\boldsymbol{s}) = \mathbb{E}_{\pi}\left[G_t \mid \boldsymbol{s}_t = \boldsymbol{s}\right], \tag{5.16}$$

and the state-action value function

$$Q^{\pi}(\boldsymbol{s}, \boldsymbol{a}) = \mathbb{E}_{\pi}\left[G_t \mid \boldsymbol{s}_t = \boldsymbol{s}, \boldsymbol{a}_t = \boldsymbol{a}\right], \tag{5.17}$$

which characterize the expected return under policy π.

In principle, Dec-POMDPs can be addressed using either model-based or model-free reinforcement learning approaches. Model-based methods require explicit knowledge or estimation of the transition dynamics $P(\boldsymbol{s}_{t+1} \mid \boldsymbol{s}_t, \boldsymbol{a}_t)$, which is infeasible in the metaverse due to high-dimensional continuous states, heterogeneous user behavior, and stochastic network conditions. Consequently, this work adopts a model-free reinforcement learning approach, which directly learns policies and value functions through interaction with the environment. The continuous action space and cooperative structure of the problem naturally motivate the use of actor-critic algorithms, which are introduced in the following section.

5.2 Multi-Agent Soft Actor-Critic with Graph Convolutional Networks

Soft Actor-Critic (SAC) is a model-free deep reinforcement learning algorithm that extends the classical actor-critic framework by explicitly incorporating an entropy maximization objective [79]. By jointly optimizing the expected cumulative reward and the stochasticity of the policy, SAC encourages sustained exploration and mitigates premature convergence to suboptimal solutions. These properties are particularly desirable in metaverse environments, where network conditions, user behavior, and computational workloads exhibit high variability and non-stationarity. In this section, we extend SAC to a cooperative multi-agent setting and further integrate graph convolutional networks to address the structural dependencies among agents. The resulting approach is referred to as SAC-GCN.

5.2.1 Multi-Agent Soft Actor-Critic

The objective of SAC-GCN is to learn a set of decentralized policies, one for each agent, such that their joint behavior maximizes the expected long-term return while maintaining sufficient exploration through entropy regularization. Formally, the optimization problem is defined as

$$\pi^* = \arg\max_{\pi} \ \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k \left(r_{t+k+1} + \alpha H(\pi(\cdot \mid \boldsymbol{s}_{t+k}))\right)\right], \tag{5.18}$$

where $H(\pi(\cdot \mid \boldsymbol{s}_t)) = \mathbb{E}_{\boldsymbol{a}}[-\log \pi(\boldsymbol{a} \mid \boldsymbol{s}_t)]$ denotes the entropy of the joint policy at state $\boldsymbol{s}_t$, and α is a temperature parameter that controls the trade-off between reward maximization and exploration.

To incorporate entropy into value estimation, SAC defines a soft state-action value function that augments the conventional return with an entropy term. Specifically, the soft Q-function is given by

$$Q(\boldsymbol{s}, \boldsymbol{a}) = \mathbb{E}_{\pi}\left[V^{\pi}(\boldsymbol{s}_{t+1}) + \alpha \sum_{k=1}^{\infty} \gamma^k H(\pi(\cdot \mid \boldsymbol{s}_{t+k}))\middle| \boldsymbol{s}_t = \boldsymbol{s}, \boldsymbol{a}_t = \boldsymbol{a}\right], \tag{5.19}$$

and the corresponding soft state value function is defined as

$$V(\boldsymbol{s}) = \mathbb{E}_{\boldsymbol{a}\sim\pi(\cdot|\boldsymbol{s})}\left[Q(\boldsymbol{s}, \boldsymbol{a}) - \alpha \log \pi(\boldsymbol{a} \mid \boldsymbol{s})\right]. \tag{5.20}$$

To extend SAC to a cooperative multi-agent setting, we adopt the centralized training and decentralized execution paradigm. During training, each agent is equipped with a centralized critic that has access to the global state and joint actions, enabling effective coordination and credit assignment. During execution, however, each agent relies solely on its local observation and learned policy, thereby preserving decentralization and scalability.

Each agent i maintains two soft Q-networks parameterized by ϕ_1^i and ϕ_2^i, together with corresponding target networks $\overline{\phi}_1^i$ and $\overline{\phi}_2^i$, in order to mitigate positive bias in value estimation and stabilize learning [80]. The target networks are updated via soft updates according to

$$\overline{\phi}_j^i \leftarrow \tau\phi_j^i + (1-\tau)\overline{\phi}_j^i, \quad j = 1, 2, \tag{5.21}$$

where $\tau \in (0, 1]$ denotes the update rate.

The policy of agent i is parameterized by θ^i, and the joint policy is denoted by $\boldsymbol{\theta} = (\theta^1, \ldots, \theta^N)$. Since execution is decentralized, the joint policy factorizes over agents as

$$\pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t \mid \boldsymbol{s}_t) = \prod_{i=1}^{N} \pi_{\theta^i}(a_t^i \mid o_t^i). \tag{5.22}$$

Policy optimization is performed by maximizing the expected soft state-action value while penalizing low-entropy actions. For each agent i, the policy objective is

$$\max_{\theta^i} \ \mathbb{E}_{\boldsymbol{a}_t \sim \pi_{\boldsymbol{\theta}}} \left[Q_\phi(\boldsymbol{s}_t, \boldsymbol{a}_t) - \alpha \log \pi_{\theta^i}(a_t^i \mid o_t^i) \right]. \tag{5.23}$$

To enable gradient-based optimization, we apply the reparameterization trick and express actions as deterministic functions of Gaussian noise,

$$\tilde{a}_t^i = \tanh\left(\mu_{\theta^i}(o_t^i) + \sigma_{\theta^i}(o_t^i) \odot \xi_t^i \right), \quad \xi_t^i \sim \mathcal{N}(0, 1), \tag{5.24}$$

where μ_{θ^i} and σ_{θ^i} denote the mean and standard deviation outputs of the policy network.

The resulting policy loss for agent i is given by

$$J_\pi(\theta^i) = \mathbb{E}_{\boldsymbol{s}_t \sim \mathcal{B}, \xi_t \sim \mathcal{N}} \left[\min_{j=1,2} Q_{\phi_j}(\boldsymbol{s}_t, \tilde{\boldsymbol{a}}_t) - \alpha \log \pi_{\theta^i}(\tilde{a}_t^i \mid o_t^i) \right], \tag{5.25}$$

where $\tilde{\boldsymbol{a}}_t = (\tilde{a}_t^1, \ldots, \tilde{a}_t^N)$ and $\mathcal{B}$ denotes the replay buffer.

The critic networks are updated by minimizing the temporal-difference loss

$$J_Q(\phi_{1,2}^i) = \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{s}_{t+1}) \sim \mathcal{B}} \left[\left(Q_{\phi_{1,2}^i}(\boldsymbol{s}_t, \tilde{\boldsymbol{a}}_t) - y' \right)^2 \right], \tag{5.26}$$

where the target value y' is computed as

$$y' = r_t + \gamma \left(\min_{j=1,2} Q_{\bar{\phi}_j}(\boldsymbol{s}_{t+1}, \tilde{\boldsymbol{a}}_{t+1}) - \alpha \log \pi_{\theta^i}(\tilde{a}_{t+1}^i \mid o_{t+1}^i) \right). \tag{5.27}$$

Through entropy-regularized policy optimization and centralized value estimation, the proposed multi-agent SAC framework enables robust and adaptive coordination under continuous state and action spaces. In the following subsection, we further enhance this framework by introducing graph convolutional networks to explicitly model inter-agent dependencies and improve scalability in dense metaverse environments.

5.2.2 Graph Convolutional Networks

In the proposed SAC-GCN framework, each agent is equipped with a centralized critic that has access to the global state and joint actions during training. While this centralized information is beneficial for coordination and credit assignment, naively concatenating observations and actions from all agents leads to scalability issues as the number of agents increases. Such an approach may introduce redundant information, hinder generalization, and obscure meaningful inter-agent dependencies. To address these limitations, we incorporate graph convolutional networks into the critic architecture to explicitly model structured interactions among agents in the metaverse.

We abstract the edge computing-enabled metaverse as an undirected interaction graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each node $v_i \in \mathcal{V}$ corresponds to an individual agent, and each edge $e_{ij} \in \mathcal{E}$ represents a potential interaction between agent i and agent j. These interactions arise from shared communication links, competition for computation resources, spatial proximity in virtual environments, or correlated user behaviors. By representing the system as a graph, inter-agent dependencies can be localized and exploited in a principled manner, rather than being implicitly learned from unstructured global inputs.

Each node in the graph is associated with a feature vector derived from the agent's local observation and recent action information, as defined in Section 5.1.3. These raw features encode both communication-related and computation-related signals that are relevant to resource allocation decisions. To project heterogeneous observations into a common latent space, an encoder network with shared parameters is applied to all agents. This encoder produces a fixed-dimensional embedding that captures the essential local state of each agent while enabling parameter sharing and improved generalization.

Following the encoding stage, a graph convolutional layer performs neighborhood aggregation based on the interaction graph. For a given agent i, its latent representa-

tion is updated by combining its own encoded features with those of its neighboring agents. This localized message-passing process allows the critic to incorporate contextual information from agents that are structurally related, while avoiding the noise introduced by distant or weakly related agents. Such a design is well aligned with metaverse environments, where resource contention and performance coupling are often spatially or topologically localized.

Formally, let $F \in \mathbb{R}^{N \times d}$ denote the matrix of encoded features for all agents, where N is the number of agents and d is the embedding dimension. The adjacency matrix $A \in \mathbb{R}^{N \times N}$ specifies the topology of the interaction graph. For each agent i, a subgraph-specific adjacency matrix A^i is constructed to include agent i and its immediate neighbors. The ordering of rows in A^i ensures that the first row corresponds to agent i, followed by its neighbors. This construction enables each critic to focus on the most relevant subset of agents while preserving a consistent input structure.

The aggregated representation used by the critic of agent i is then computed as

$$H^i = \mathrm{GCN}(A^i F), \tag{5.28}$$

where the GCN operation propagates and transforms information along graph edges to produce a joint embedding that captures spatially correlated interactions. This embedding serves as the input to the centralized Q-network of agent i. To further enhance representational flexibility, the aggregation process is augmented with a self-attention mechanism, which adaptively weights neighboring agents according to their contextual relevance. The attention mechanism is detailed in the following subsection.

Figure 5.1 illustrates the overall architecture of SAC-GCN. Each critic consists of an encoder that transforms raw observations into latent embeddings, a graph convolutional module that aggregates information from connected agents, and a centralized Q-network that estimates soft state-action values based on the aggregated representation. During execution, only the decentralized actor is used, and decisions are made solely based on local observations.

By embedding graph convolutional networks within the centralized critic, the

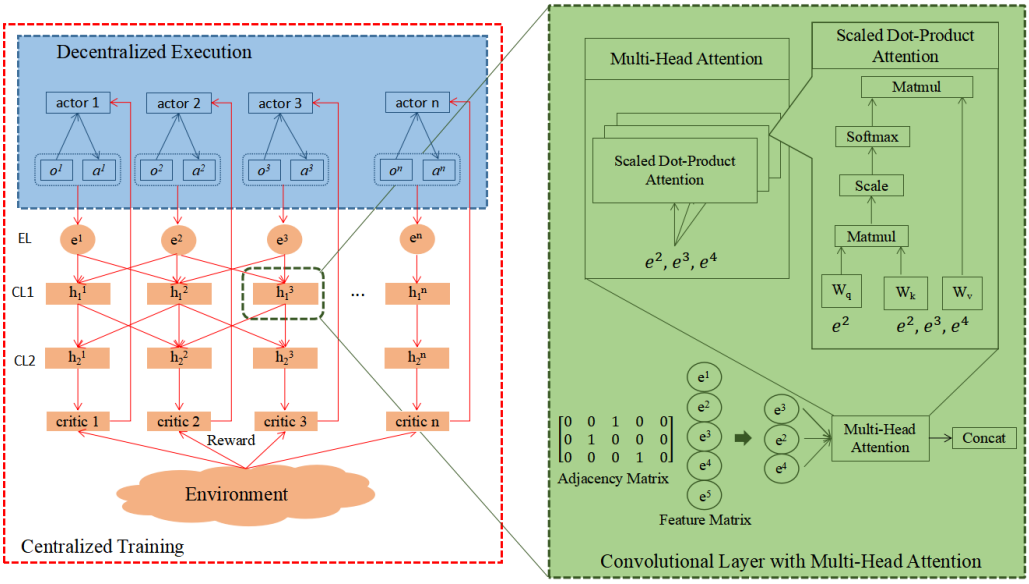


Figure 5.1: Neural network architecture of SAC-GCN. EL and CL denote the encoder layer and the graph convolutional layer, respectively. During training, each agent is equipped with a centralized critic that processes joint states and actions, while during execution, a decentralized actor selects actions based solely on local observations. Graph convolutional networks with multi-head attention are employed to generate the final inputs to the critic.

proposed approach achieves scalable coordination, improved credit assignment, and stronger generalization in dense multi-agent environments. Moreover, this modular architecture naturally accommodates dynamic interaction patterns and partial observability, both of which are fundamental characteristics of large-scale metaverse systems.

5.2.3 Self-Attention Mechanism

While graph convolutional networks enable localized information aggregation based on the interaction topology, they implicitly assume that all neighboring agents contribute equally to the representation of a target agent. In metaverse environments, this assumption is often violated. The impact of one agent's action on another can vary significantly depending on shared network links, contention for computation resources, or spatial proximity within a virtual scene. To capture such heterogeneous inter-agent influences, we further integrate a self-attention mechanism into the graph convolutional layer of the centralized critic.

The core idea of the self-attention mechanism is to allow each agent to adaptively weight the contributions of its neighboring agents when aggregating features. This enables the critic to focus on the most influential neighbors while suppressing irrelevant or weakly coupled interactions. Such selective aggregation is particularly important in resource allocation problems, where actions taken by one user, such as increasing transmission bit rate, may strongly affect a subset of nearby users but have negligible impact on others.

The self-attention mechanism operates on the latent feature embeddings produced by the shared encoder. For a target agent i with encoded feature vector e^i, a query representation is first constructed as

$$Q = W_q e^i, \tag{5.29}$$

where W_q is a learnable projection matrix. For each neighboring agent j, its encoded

Algorithm 5.1 SAC-GCN

Initialize two soft Q-networks with parameters ϕ_1^i, ϕ_2^i and a policy network with parameters θ^i, for each agent i
Initialize two target networks with $\overline{\phi_1^i} \leftarrow \phi_1^i$ and $\overline{\phi_2^i} \leftarrow \phi_2^i$, for each agent i
for episode $e = 1, 2, ...$ **do**
  for time step $t = 1, 2, ..., T$ **do**
    Obtain the original observation o_t^i for each agent and current global state $\boldsymbol{s}_t$
    for each agent i **do**
      Take an action $\tilde{a}_t^i$ generated by Eq. 5.24
      Receive shared reward r_t and observe o_{t+1}^i
    end for
    Obtain joint action $\tilde{\boldsymbol{a}}_t$, and next global state $\boldsymbol{s}_{t+1}$
    Store the tuple sample $(\boldsymbol{s}_t, \tilde{\boldsymbol{a}}_t, r_t, \boldsymbol{s}_{t+1})$ into the replay buffer $\mathcal{B}$
    for each agent i **do**
      Get a random batch of samples from $\mathcal{B}$
      Apply attention mechanism (Eq. 5.33) to get the inputs of each critic h_t and h_{t+1} based on global states $\boldsymbol{s}_t$ and $\boldsymbol{s}_{t+1}$
      Update Q-networks by minimizing the loss function (Eq. 5.26) with $\boldsymbol{s}_t$ replaced by h_t and $\boldsymbol{s}_{t+1}$ replaced by h_{t+1}
      Update policy networks by maximizing the target function (Eq. 5.25) with $\boldsymbol{s}_t$ replaced by h_t
      Update target Q-networks by Eq. 5.21
    end for
  end for
end for

feature e^j is projected into a key vector and a value vector according to

$$K^j = W_k e^j, \quad V^j = W_v e^j, \tag{5.30}$$

where W_k and W_v are also learnable parameters shared across agents.

The relevance between the target agent and each neighbor is quantified using cosine similarity between the query and key representations,

$$S(Q, K^j) = \frac{Q \cdot K^j}{\|Q\| \cdot \|K^j\|}. \tag{5.31}$$

These similarity scores are normalized through a softmax operation to produce attention weights,

$$\alpha^j = \frac{\exp(S(Q, K^j))}{\sum_j \exp(S(Q, K^j))}, \tag{5.32}$$

which reflect the relative importance of each neighboring agent to the target agent. The final attention output is obtained as a weighted aggregation of value vectors,

$$\text{Attention}(Q, K, V) = \sum_j \alpha^j V^j. \tag{5.33}$$

To further enhance representational capacity, we adopt a multi-head attention mechanism, where multiple sets of projection matrices are used to compute attention in parallel. Each attention head captures a distinct pattern of inter-agent dependency, and the resulting outputs are concatenated to form a comprehensive neighborhood representation. This design allows the centralized critic to model diverse interaction modalities, such as competition for bandwidth and contention for computation resources, within a unified framework.

By integrating self-attention into the graph convolutional layer, the centralized critic gains the ability to dynamically prioritize critical neighbors and perform more accurate value estimation under complex interaction patterns. This attention-enhanced aggregation improves coordination among agents, accelerates convergence during training, and enhances generalization to varying network and workload conditions.

A complete description of the SAC-GCN algorithm, including graph-based aggregation, multi-head self-attention, and actor-critic updates, is provided in Algorithm 5.1. At each training iteration, transitions are sampled from the replay buffer

$\mathcal{B}$, and the centralized critic computes aggregated representations using the attention mechanism defined in Eq. 5.33. Network parameters are then updated following the centralized training and decentralized execution paradigm. After training, only decentralized actors are retained for execution, and each agent selects actions solely based on its local observation. As a result, the deployment of SAC-GCN introduces no additional communication overhead or inference latency, making it well suited for real-time resource allocation in latency-sensitive metaverse environments.

5.3 Experiment and Analysis

This section presents a comprehensive experimental evaluation of the proposed SAC-GCN framework for dynamic and intelligent resource allocation in metaverse environments. The objective is to assess the framework's ability to jointly manage communication and computation resources under realistic and time-varying conditions, and to quantify its effectiveness in improving user-perceived QoE. Rather than focusing on a synthetic benchmark, the evaluation is conducted on a system-level prototype that integrates real-time rendering, network streaming, and adaptive control mechanisms.

5.3.1 System Setup

To evaluate the proposed framework in a realistic metaverse setting, we implement an end-to-end experimental platform that integrates immersive rendering, real-time communication, and adaptive resource control. The hardware infrastructure consists of a high-performance desktop workstation running Windows 10, equipped with an Intel Core i9-11900F CPU, 64 GB of RAM, and an NVIDIA GeForce RTX 3090 GPU. This configuration provides sufficient computational capacity to support concurrent rendering tasks and fine-grained resource management for multiple users.

The metaverse environment is developed using Unreal Engine, a widely adopted real-time 3D rendering platform. Multiple Unreal Engine instances are launched

concurrently on the rendering server, with each instance generating an independent viewpoint for an individual user. The server is connected via Ethernet to a TP-Link TL-WR1043ND Wi-Fi router to ensure stable upstream connectivity. Rendered video streams are delivered to Meta Quest 2 headsets using the Unreal Pixel Streaming framework, which is built on WebRTC to support low-latency and high-fidelity media transmission [81].

Within the WebRTC-based streaming pipeline, user interaction signals and motion data are transmitted upstream as low-bandwidth control flows, whereas high-resolution rendered frames are transmitted downstream as bandwidth-intensive video flows. This asymmetric traffic pattern is characteristic of XR streaming systems. To enable adaptive control, two resource management modules are deployed, namely an adaptive bit rate server and an adaptive CPU usage server. These modules communicate with Unreal Engine instances via Socket.IO, enabling real-time bidirectional signaling. The ABR server adjusts the target transmission bit rate based on network feedback, including round-trip latency, jitter, and packet loss. In parallel, the ACU server regulates CPU consumption using the BES process governor to throttle rendering workloads at the per-instance level.

To emulate diverse network conditions, we employ Clumsy, a network emulation tool that allows controlled manipulation of latency, bandwidth, and packet loss. This setup enables systematic evaluation under both favorable and adverse networking scenarios. The default streaming configuration includes a total downlink bandwidth of 300 Mbps, an average latency of 20 ms, a packet loss rate of 0.5%, a video resolution of 2048×1080, a target frame rate of 60 fps, and an upper bound of 80% CPU utilization for rendering tasks. To assess adaptability, key system parameters are varied over wide ranges, including network delay, packet loss rate, available bandwidth, and CPU capacity.

Table 5.1 summarizes a set of representative network scenarios used throughout the experiments, covering both stable and highly dynamic conditions.

5.3.2 Parameter Optimization of the QoE Model

To calibrate the penalty coefficients $(\alpha, \beta, \gamma, \delta)$ in the proposed QoE formulation in Eq. 5.10, and to validate its consistency with subjective user perception, we conduct a controlled user study involving eight volunteers. The purpose of this study is to establish a quantitative mapping between system-level metrics and perceived immersive quality under realistic metaverse workloads.

Table 5.1: Representative Network Conditions for Metaverse Applications

Scenario	Bandwidth	Latency	Jitter	Packet Loss	Burst Loss
S1	100–200	10–30	2–5	0.1	0
S2	50–100	30–50	5–10	0.5	0.5
S3	20–80	50–100	10–20	1	1
S4	200–500	5–10	1–3	0.1	0
S5	100→30	50→100	5→20	0.5→5	10
S6	30→100	100→20	20→5	2→0.5	5

Six representative streaming scenarios are designed to capture diverse deployment conditions, including stable high-performance rendering, home Wi-Fi streaming, mobile network access, edge-assisted rendering, congestion-induced degradation, and post-degradation recovery. The detailed network characteristics of these scenarios are summarized in Table 5.1. Each participant experiences every scenario during a fixed 5-minute session using a VR headset and subsequently rates their experience using a Mean Opinion Score on a five-point Likert scale.

To reduce subjective bias and day-specific effects, each participant repeats all six scenarios four times on different days, resulting in 192 total evaluations. MOS scores are averaged across repetitions and participants to obtain a stable reference QoE value for each scenario. A grid search procedure is then applied to identify the set of coefficients that minimizes the root mean square error between predicted QoE values and averaged MOS scores. The resulting optimal coefficients are $\alpha = 1.0$, $\beta = 0.3$,

$\gamma = 0.4$, and $\delta = 0.2$.

With these parameters, the QoE model achieves a coefficient of determination of $R^2 = 0.92$, indicating strong agreement with subjective user assessments. A sensitivity analysis further confirms robustness, with an average RMSE variation of 5.2% under independent $\pm 20\%$ perturbations of each coefficient. These results demonstrate that the proposed QoE model is both accurate and stable, making it suitable for integration into adaptive resource allocation algorithms.

5.3.3 Evaluation Baselines

To benchmark the effectiveness of SAC-GCN, we compare it against several representative baseline methods that span different algorithmic paradigms, including classical congestion control and reinforcement learning-based approaches.

- Deep Q-Network (DQN): A value-based reinforcement learning method that approximates Q-functions using deep neural networks [82]. DQN is treated as a single-agent baseline without explicit multi-agent coordination.
- Independent Soft Actor-Critic (ISAC): A decentralized variant in which each agent independently learns a SAC policy while treating other agents as part of the environment. ISAC is simple and scalable but suffers from non-stationarity in cooperative settings.
- Google Congestion Control with Greedy Computation (GCC-G): A WebRTC-native congestion control algorithm that adapts transmission rates based on delay and loss signals [83], combined with unrestricted CPU usage.
- Bottleneck Bandwidth and Round-trip Time with Greedy Computation (BBR-G): A model-based congestion control protocol that estimates bottleneck bandwidth and minimum RTT, paired with unregulated computation resource usage.

These baselines collectively represent state-of-the-art industrial solutions and widely used reinforcement learning approaches. The comparative analysis highlights the benefits of integrating cooperative learning, structured representation, and joint communication-computation control in complex metaverse environments.

5.3.4 Offline Training

As discussed earlier, each agent in the SAC-GCN architecture, illustrated in Fig. 5.1, is equipped with a composite neural network structure consisting of two encoder layers and a graph convolutional layer, in addition to its actor and critic networks. This modular design enables effective feature abstraction from local observations while supporting structured information exchange among neighboring agents. Both the actor and soft Q-networks adopt a standard feedforward architecture composed of an input layer, two fully connected hidden layers with ReLU activation, and an output layer. Such a configuration provides sufficient expressive capacity for modeling policies and value functions in continuous and high-dimensional action spaces.

For reproducibility and fair comparison, all reinforcement learning baselines, including SAC-GCN, ISAC, and DQN, are implemented using identical network architectures wherever applicable and share the same set of hyperparameters. A detailed summary of these configurations is provided in Table 5.2. Unlike episodic tasks with clearly defined terminal states, metaverse interactions are continuous and open-ended. To enable stable training and consistent evaluation, we empirically define a fixed temporal window of 40 seconds as one episode. Agents make decisions at 1-second intervals, resulting in 40 decision steps per episode. This design reflects time-constrained streaming workloads while preserving sufficient temporal resolution for adaptive control.

All learning-based methods are trained over multiple episodes under diverse network conditions and rendering workloads, as described in Section 5.3. This training protocol exposes each algorithm to a wide range of environmental dynamics and pre-

Table 5.2: Parameter Specifications in the Experiments

Parameters	**Value**
Optimizer	Adam
Number of neurons in hidden layer	128
Reward discount factor	0.99
Replay buffer size	5000
Learning rate for updating Q-networks	0.0001
Learning rate for updating policy network	0.0005
Batch size	64
Coefficient for updating target Q-networks	0.01
Entropy temperature parameter	0.2
Metaverse scene satisfaction level factor	1
Choppiness penalty factor	0.2
Latency penalty factor	0.05
Instability penalty factor	0.5
Overall QoE factor	2
Communication resource balancing factor	-0.6
Computation resource balancing factor	-0.6
Number of users in the metaverse	5

vents overfitting to static configurations. Fig. 5.2 presents the cumulative reward trajectories of the three reinforcement learning approaches during offline training. At early stages, all methods exhibit similarly low reward levels, which is expected given the lack of prior experience and the high degree of uncertainty in joint resource allocation. DQN, in particular, shows slower initial progress due to its reliance on value estimation without explicit exploration incentives.

As training proceeds, SAC-GCN and ISAC demonstrate markedly faster reward growth than DQN. This improvement is primarily attributed to the entropy-regularized exploration mechanism in SAC, which encourages diverse action sampling and mitigates premature convergence. In contrast, DQN employs deterministic action selection and struggles to explore effectively in a continuous and stochastic action space. More importantly, SAC-GCN consistently outperforms ISAC in the later stages of training. This advantage arises from the centralized training and decentralized execution paradigm adopted by SAC-GCN. The centralized critic exploits global, graph-structured information to guide policy updates, while the GCN layers enable agents to leverage contextual features from neighboring users. ISAC, by treating other agents as part of the environment, lacks explicit mechanisms for inter-agent reasoning and suffers from suboptimal credit assignment. By the end of training, SAC-GCN achieves the highest cumulative reward among all methods, indicating its superior ability to learn cooperative, robust, and generalizable policies for dynamic metaverse environments.

5.3.5 Overall User Quality of Experience Analysis

We first examine the effectiveness of each resource allocation strategy in enhancing overall user QoE under varying network delay conditions, as shown in Fig. 5.3(a). The delay in this experiment is artificially introduced prior to transmission and does not include inherent propagation or queuing delays. Across all methods, QoE consistently declines as delay increases, with particularly sharp degradation under

high-delay regimes. This behavior is primarily driven by increased motion-to-photon latency, which amplifies visual inconsistency and induces user discomfort, a phenomenon widely observed in immersive XR systems.

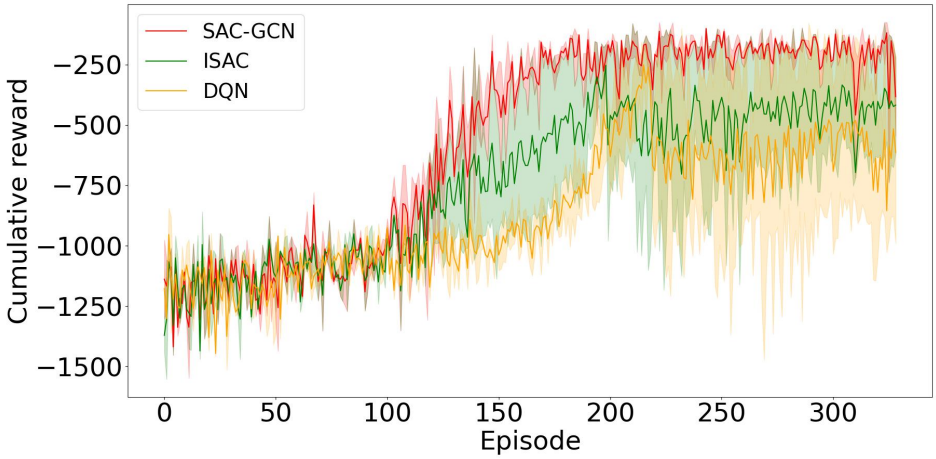
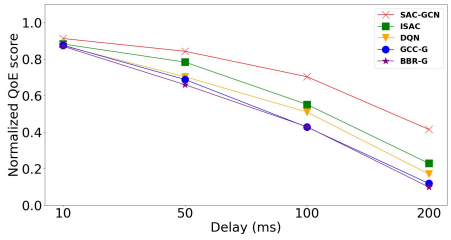


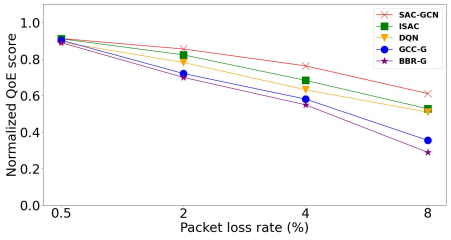
Figure 5.2: Cumulative reward versus episode (40 s per episode) for different DRL methods during the offline training process.

Among all methods, SAC-GCN exhibits the strongest resilience to increasing delay. By leveraging attention-enhanced graph convolution, agents can selectively prioritize information from neighbors experiencing elevated latency and cooperatively adjust bit rate decisions. At a delay of 100 ms, SAC-GCN improves overall QoE by 27%, 37%, 63%, and 64% compared with ISAC, DQN, GCC-G, and BBR-G, respectively. These gains highlight the effectiveness of context-aware coordination enabled by structured inter-agent modeling.

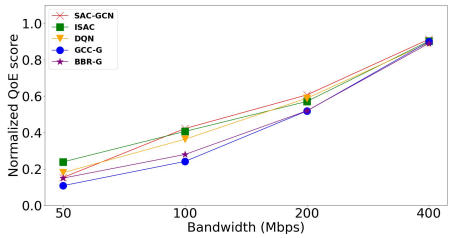
The impact of packet loss on QoE is illustrated in Fig. 5.3(b). As packet loss increases, all methods experience performance degradation due to increased frame drops and playback instability. BBR-G is particularly vulnerable under lossy conditions, showing pronounced QoE deterioration as loss rates exceed 2%. In contrast, SAC-GCN maintains consistently higher QoE across all loss levels. This robustness stems from its ability to infer shared congestion patterns through neighbor-aware aggregation, enabling proactive bit rate adaptation and improved temporal stability.



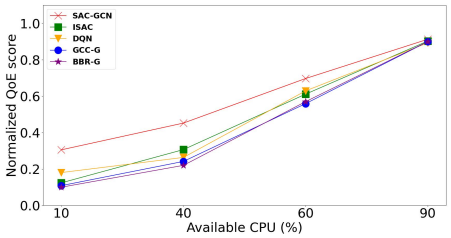
(a) Delays (10 ms, 50 ms, 100 ms, 200 ms)



(b) Packet loss rates (0.5%, 2%, 4%, 8%)



(c) Bandwidth levels (50 Mbps, 100 Mbps, 200 Mbps, 400 Mbps)



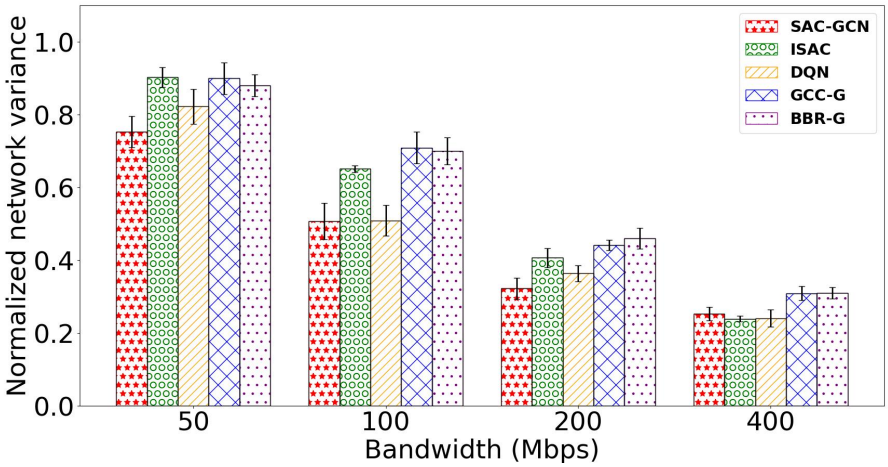
(d) Available CPU (10%, 40%, 60%, 90%)

Figure 5.3: Overall QoE score comparison of five resource allocation methods under different network and system conditions.

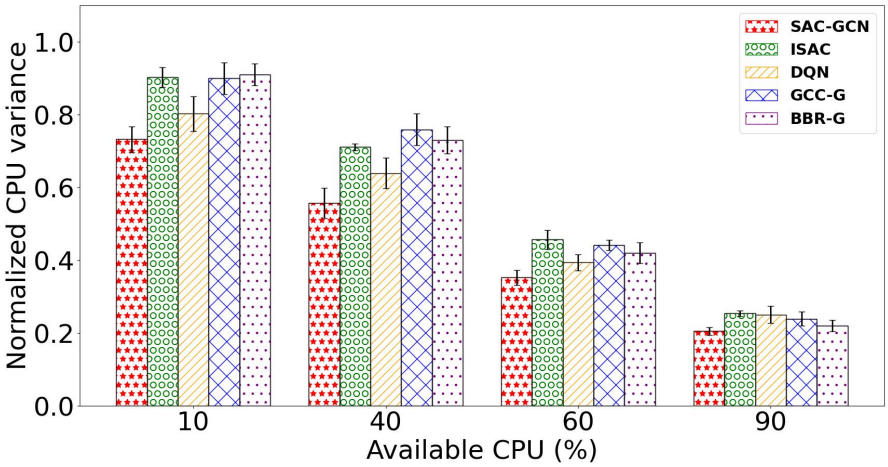
Fig. 5.3(c) evaluates QoE under varying bandwidth availability. When bandwidth is severely constrained, all methods suffer from significant QoE loss due to insufficient transmission capacity. Under extreme scarcity, ISAC marginally outperforms SAC-GCN, as its fully decentralized operation avoids coordination overhead. However, as bandwidth increases, the advantages of SAC-GCN become increasingly evident. With sufficient resources, SAC-GCN achieves near-optimal QoE and consistently outperforms all baselines, confirming that structured coordination is most beneficial when communication constraints are moderate rather than extreme.

5.3.6 Resource Allocation Balancing Analysis

We next assess fairness in resource allocation by examining the variance of communication and computation resources across users. Lower variance indicates more balanced and equitable allocation.



(a) Impact of network bandwidth on communication resource balance



(b) Impact of available CPU on computation resource balance

Figure 5.4: Comparison of resource allocation balancing under different settings.

For communication resources, Fig. 5.4(a) shows that variance decreases as total bandwidth increases across all methods. SAC-GCN consistently achieves the lowest variance, reflecting its ability to redistribute bandwidth equitably through centralized learning and attention-based neighbor modeling. DQN also benefits from centralized value estimation, whereas ISAC, GCC-G, and BBR-G exhibit higher variance due to their decentralized or greedy nature.

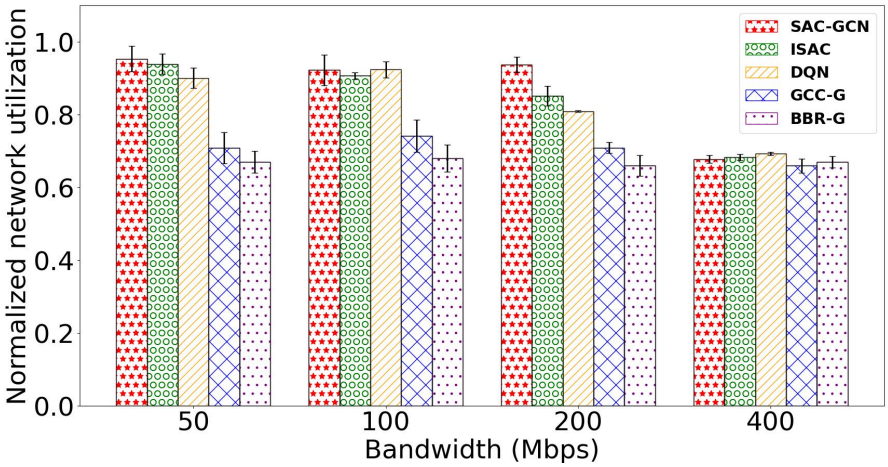
For computation resources, similar trends are observed in Fig. 5.4(b). Under constrained CPU availability, methods lacking coordination allow certain users to monopolize resources. SAC-GCN and DQN demonstrate superior balancing performance, while GCC-G and BBR-G show persistent imbalance due to unrestricted CPU usage. SAC-GCN's graph-based critic further improves fairness by explicitly modeling inter-agent computation contention.

5.3.7 Resource Utilization Rate Analysis

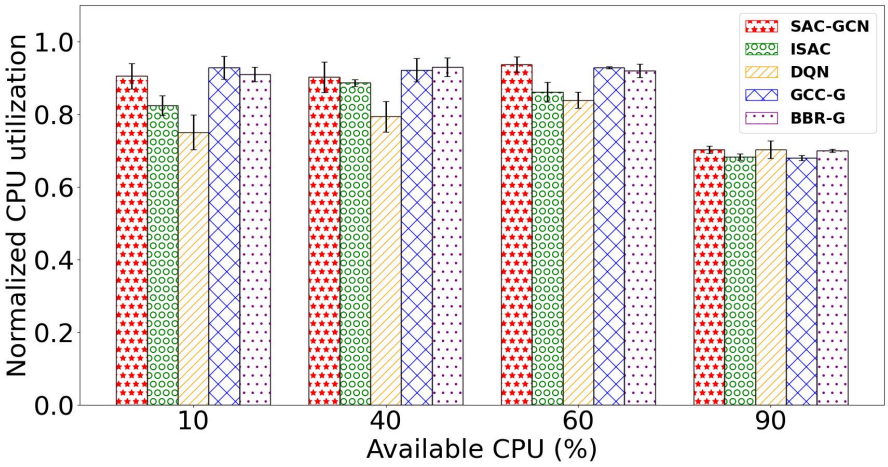
Finally, we evaluate resource utilization efficiency. Fig. 5.5(a) reports communication utilization under varying bandwidth conditions. While utilization is unstable under severe constraints, SAC-GCN consistently achieves equal or higher utilization as resources become sufficient. At 200 Mbps, SAC-GCN improves utilization by 10%, 15%, 32%, and 41% over ISAC, DQN, GCC-G, and BBR-G, respectively.

Fig. 5.5(b) illustrates computation utilization under varying CPU capacity. Greedy methods achieve high utilization under constrained conditions but at the cost of severe imbalance. SAC-GCN achieves a more desirable trade-off, maintaining high utilization while preserving fairness. At 60% CPU availability, SAC-GCN outperforms ISAC and DQN by 8% and 12%, respectively.

Overall, these results demonstrate that SAC-GCN achieves a balanced combination of high QoE, equitable resource allocation, and efficient utilization, validating its suitability for real-time, multi-user metaverse environments.



(a) Network resource utilization



(b) CPU resource utilization

Figure 5.5: Comparison of the five resource allocation methods in terms of resource utilization rates.

5.3.8 Discussion

The experimental results presented above provide several important insights into the effectiveness of deep reinforcement learning based approaches for resource allocation in metaverse environments, as well as the design principles that underpin the proposed

SAC-GCN framework.

First, the consistent superiority of DRL-based methods over conventional congestion control mechanisms highlights the fundamental limitations of rule-based adaptation in complex, multi-user metaverse systems. Traditional congestion control algorithms such as GCC and BBR operate on reactive heuristics that respond to isolated network signals, without explicit consideration of user-level experience or system-wide objectives. As a result, their performance degrades rapidly under adverse conditions such as high latency or packet loss. In contrast, DRL agents learn policies through continuous interaction with the environment and directly optimize a reward function aligned with user-perceived QoE. This learning-based paradigm enables proactive and fine-grained adjustment of both communication and computation resources, allowing the system to adapt more effectively to non-stationary and heterogeneous operating conditions.

Second, the performance gap between ISAC and SAC-GCN underscores the importance of explicit coordination in multi-agent resource allocation. Although both methods are based on entropy-regularized reinforcement learning, ISAC relies entirely on decentralized learning and treats other agents as part of a non-stationary environment. This independence assumption limits its ability to capture inter-agent dependencies and often leads to inefficient resource sharing and large allocation variance under system stress. SAC-GCN, by contrast, adopts a centralized training and decentralized execution paradigm, enabling agents to exploit global information during learning while preserving scalability at runtime. This structure significantly improves credit assignment and aligns local decision-making with system-level objectives.

Third, the integration of graph convolutional networks and self-attention mechanisms plays a critical role in enhancing coordination efficiency. The graph-based abstraction allows the centralized critic to explicitly model structural relationships among agents, such as shared bandwidth links or computation contention, rather than implicitly inferring these dependencies from unstructured global inputs. The self-attention mechanism further refines this process by enabling the critic to priori-

tize influential neighbors based on contextual relevance. Together, these components allow SAC-GCN to focus learning capacity on the most impactful interactions, resulting in more balanced resource allocation and improved robustness without sacrificing overall QoE.

Taken together, these findings suggest that effective resource allocation in metaverse environments requires not only learning-based adaptation, but also architectural designs that explicitly encode inter-agent structure and coordination mechanisms. The proposed SAC-GCN framework demonstrates that combining centralized learning, graph-based representation, and attention-driven aggregation provides a principled and scalable solution for managing communication and computation resources in large-scale, agentic metaverse systems.

5.4 Limitations

Although this chapter demonstrates the effectiveness of the proposed SAC-GCN framework for adaptive resource allocation in metaverse environments, several limitations should be acknowledged.

First, the experimental evaluation is conducted on a specific prototype system with a limited deployment scale. The testbed is built using Unreal Engine, WebRTC-based Pixel Streaming, Meta Quest 2 headsets, and a single rendering server equipped with an NVIDIA RTX 3090 GPU. This platform provides a realistic basis for studying multi-user metaverse streaming and edge-side resource allocation. However, the experimental setting still differs from a large-scale practical edge-cloud deployment. The number of users is limited, and all rendering tasks are managed within a controlled server environment rather than across multiple geographically distributed edge nodes. In real metaverse systems, users may connect through heterogeneous access networks, move across edge regions, and share infrastructure with many other services. These factors may introduce additional challenges such as inter-edge handover, heterogeneous server capacity, cross-region scheduling, and large-scale GPU contention.

Therefore, although the results validate the feasibility of SAC-GCN in a prototype environment, further experiments under larger-scale and more heterogeneous edge-cloud deployments are needed to confirm its practical generality.

Second, the QoE model and reward function rely on simplified and fixed representations of user experience. In this chapter, the reward function jointly considers visual quality, frame smoothness, latency, quality stability, and resource balancing. The QoE coefficients are calibrated through a user study involving eight participants under six representative network scenarios. This design provides an empirical basis for aligning system-level metrics with subjective perception. However, user experience in immersive metaverse applications is highly individual and context-dependent. Different users may have different sensitivity to latency, visual quality fluctuation, motion sickness, interaction delay, and rendering artifacts. In addition, the MOS-based study mainly captures short-term perceived quality, while long-term fatigue, cybersickness, task success, and social interaction quality are not fully modeled. The fixed reward weights may therefore be insufficient to represent personalized QoE preferences or diverse application requirements. Future work should incorporate larger-scale user studies, personalized QoE modeling, and adaptive reward mechanisms that can better reflect heterogeneous user perception.

Third, the reinforcement learning process is mainly based on offline training under controlled and emulated network conditions. The use of Clumsy enables systematic manipulation of latency, bandwidth, and packet loss, which is useful for comparing different algorithms under repeatable scenarios. However, real metaverse networks are often more complex and non-stationary. Network conditions may change abruptly due to user mobility, wireless interference, server-side congestion, and competing background traffic. In such environments, a policy trained offline may encounter unseen states or workload patterns that differ from the training distribution. Moreover, online exploration in real XR systems may negatively affect user experience if agents select unstable or unsafe resource allocation actions during learning. Therefore, future work should investigate safer and more adaptive learning strategies, such as offline-to-

online fine-tuning, constrained reinforcement learning, transfer learning, and robust policy adaptation under real-world network dynamics.

Fourth, the coordination mechanism of SAC-GCN depends on the selected graph structure, observation features, and centralized training design. The proposed method uses graph convolutional networks and self-attention to model inter-agent dependencies and improve cooperative resource allocation. This design is effective for capturing structured relationships among users, such as shared bandwidth links, computation contention, or spatial proximity. However, the construction of the interaction graph may strongly influence the learned policy. If the graph fails to accurately represent actual resource coupling among users, the critic may aggregate irrelevant information or ignore important dependencies. In addition, centralized critics require access to global state information during training, which may introduce scalability, synchronization, and privacy challenges as the number of users increases. Although decentralized execution avoids additional runtime communication overhead, training large-scale agent populations with dynamic graph structures remains challenging. Future work should explore adaptive graph construction, privacy-preserving training, and scalable multi-agent learning methods for larger and more dynamic metaverse systems.

Chapter 6

Multimodal Benchmarking for Metaverse Scene Understanding

6.1 Metaverse Understanding Dataset Construction

To systematically assess the scene understanding performance of Multimodal Large Language Models (MLLMs) within metaverse settings, this study develops a purpose-built dataset tailored to the perceptual and semantic characteristics of immersive virtual environments. Unlike conventional image benchmarks derived from natural scenes or web images, metaverse environments exhibit synthetic visual patterns, complex spatial layouts, persistent digital objects, and continuous human avatar interactions. These properties necessitate a dedicated dataset that faithfully reflects the structural, interactive, and dynamic features of metaverse scenes.

The dataset is constructed based on real metaverse applications, such as Roblox, Minecraft, and VRChat. The selected environment represents a prototypical urban public space and integrates a rich combination of visual and semantic elements. Static components include pedestrian walkways, open plazas, architectural structures, street furniture, signage, and diverse vegetation types. These elements form heterogeneous spatial configurations with varying depth, occlusion relationships, and geometric complexity. In addition to static infrastructure, the environment incorporates dynamic

entities such as human avatars, service robots, bicycles, and vehicles, enabling the observation of motion patterns, social interactions, and multi-agent coexistence within shared spaces. The coexistence of static and dynamic elements creates a visually dense and semantically layered scene that is well suited for evaluating advanced scene understanding capabilities.

Multiple avatars are concurrently instantiated and controlled to emulate common user behaviors, including walking, stopping, group interaction, object observation, and navigation across different regions of the environment. Camera viewpoints are configured to mirror typical metaverse perspectives, including first-person user views, slightly elevated over-the-shoulder views, and oblique viewpoints that arise during navigation and interaction. These viewpoints are dynamically adjusted in response to avatar motion and interaction context, ensuring that the collected visual data reflect authentic metaverse observation conditions rather than static or artificially staged perspectives.

Rendered visual data preserves high spatial resolution, accurate color representation, and fine-grained texture details, all of which are critical for evaluating object recognition, spatial reasoning, and interaction understanding. Temporal coherence is maintained by sampling frames from continuous rendering sequences rather than isolated screenshots, thereby retaining realistic motion cues and contextual continuity across adjacent frames.

From the simulated sessions, a total of 400 frames are uniformly sampled to construct the final dataset. Uniform sampling is adopted to balance coverage across different scene regions, interaction states, and motion conditions while avoiding overrepresentation of visually similar frames. The selected frames span a wide range of spatial layouts, including open areas, narrow pathways, and densely populated regions. They also capture diverse object configurations, varying degrees of crowd density, and multiple stages of agent interaction, such as approaching, engaging, and dispersing. Motion states range from static standing postures to complex multi-agent movement patterns, providing rich visual evidence for evaluating temporal and rela-

tional reasoning.

The resulting dataset offers a structured yet diverse representation of metaverse scenes that integrates geometric complexity, semantic richness, and interaction dynamics. By grounding data collection in realistic multi-user simulations and high-quality rendering outputs, the dataset establishes a robust foundation for benchmarking multimodal scene understanding models in immersive virtual environments.

6.2 Representative MLLMs

To comprehensively benchmark multimodal scene understanding in metaverse environments, this study evaluates a set of 13 representative MLLMs for generating scene-level descriptions for each frame in the constructed dataset. The selection of models is not intended to be exhaustive, but rather to reflect the current landscape of multimodal modeling along two key dimensions. The first dimension concerns practical influence and adoption, prioritizing models that are widely used in either industrial deployments or academic research. The second dimension concerns architectural and design diversity, ensuring that the benchmark captures heterogeneous approaches to multimodal fusion, visual encoding, and cross-modal reasoning.

Based on these considerations, the selected models are organized into three broad categories that reflect differences in openness, scale, and design objectives. This categorization facilitates structured comparison and enables analysis of how architectural choices and resource assumptions influence metaverse-oriented scene understanding performance.

The first category consists of closed-source frontier models, represented by GPT-5-mini and Gemini-2.5-Pro. These models are developed by major industrial actors and embody the current state of proprietary multimodal systems. They typically integrate large-scale vision encoders with powerful language backbones and benefit from extensive pretraining on diverse multimodal corpora. As a result, they are expected to exhibit strong visual perception, robust commonsense reasoning, and fluent

natural language generation. Their inclusion in the benchmark serves two purposes. First, they establish an upper-bound reference for multimodal scene understanding performance under unconstrained resource settings. Second, they provide a practical point of comparison for evaluating the performance gap between closed-source frontier systems and open-source alternatives in complex metaverse scenes.

The second category includes large-scale open-source multimodal models, such as Qwen2-VL-72B and LLaVA-v1.5-13B. These models adopt high-capacity architectures with tens of billions of parameters or strong visual-language alignment mechanisms, enabling detailed object grounding and nuanced semantic reasoning. Unlike proprietary models, these systems are publicly accessible and reproducible, making them particularly attractive for academic research and methodological analysis. Their architectures typically combine pretrained vision encoders with instruction-tuned language models and employ explicit cross-modal projection layers or attention mechanisms to align visual and textual representations. By incorporating these models, the benchmark evaluates whether open-source systems can achieve competitive scene understanding performance in immersive virtual environments while maintaining transparency and reproducibility.

The third category focuses on lightweight open-source models, exemplified by LLaVA-v1.5-7B and Yi-VL-6B. These models are designed with parameter efficiency and deployment feasibility in mind, often targeting scenarios with limited computational resources. Although their representational capacity is lower than that of large-scale counterparts, they offer valuable insights into the trade-offs between model size, inference cost, and multimodal understanding quality. In the context of metaverse applications, such models are particularly relevant for edge or near-edge deployment, where real-time responsiveness and resource constraints are critical considerations. Including this category enables a more nuanced assessment of how well compact multimodal models can capture spatial structure, interactions, and environmental semantics in visually complex scenes.

For each frame in the dataset, all selected MLLMs are tasked with generating a

scene-level textual description. To ensure consistency and comparability, an identical instruction prompt is used across all models. The prompt explicitly requests a holistic interpretation of the scene, including the identification of salient objects, their spatial relationships, observed actions or interactions, and relevant environmental attributes. By adopting a unified prompt formulation, the evaluation minimizes confounding effects arising from prompt engineering and focuses the comparison on intrinsic model capabilities.

All models perform inference independently under standardized settings, with temperature and decoding strategies fixed within each model's recommended configuration. No model-specific post-processing or response filtering is applied. This controlled evaluation protocol ensures that observed differences in generated descriptions can be attributed to variations in perceptual sensitivity, spatial reasoning ability, and semantic organization, rather than differences in prompting or inference procedures.

The inclusion of closed-source frontier models, large-scale open-source models, and lightweight open-source models enables a balanced and informative benchmark. This design supports both absolute performance comparison and relative analysis across different model classes, providing a comprehensive view of current multimodal scene understanding capabilities in metaverse environments.

6.3 MLLM Evaluation Protocol Design

Evaluating multimodal scene understanding in metaverse environments poses fundamental methodological challenges. Unlike conventional vision benchmarks that rely on object-level labels or short captions, metaverse scenes are characterized by complex spatial organization, persistent digital entities, and multi-agent interactions. Scene descriptions are therefore inherently open-ended, making it difficult to define a single ground-truth reference. To address these challenges, we design an evaluation protocol that emphasizes semantic completeness, relational reasoning, and interpretive coherence, while remaining robust to linguistic variation across models.

6.3.1 Semantic Evaluation Dimensions

We define four complementary semantic dimensions to jointly capture perceptual accuracy and scene-level reasoning ability. These dimensions are designed to reflect the core cognitive requirements of metaverse understanding rather than isolated visual recognition performance.

The first dimension, object recognition, evaluates whether a model correctly identifies salient entities that are essential for interpreting the scene. This includes static environmental components such as buildings, pathways, vegetation, and street furniture, as well as dynamic agents such as avatars, vehicles, and other interactive entities. The evaluation does not reward exhaustive listing of all visible objects. Instead, emphasis is placed on recognizing functionally relevant elements that contribute to the semantic structure of the scene, such as objects that shape navigation, interaction, or social context.

The second dimension, spatial reasoning, assesses the model's ability to describe meaningful spatial relationships among entities. This includes relative position, distance, orientation, and containment relationships, as well as more abstract spatial concepts such as grouping and separation. A high-quality description should be geometrically consistent with the underlying scene layout and avoid contradictory or physically implausible spatial claims. This dimension captures whether the model can move beyond isolated object identification to construct a structured spatial representation of the environment.

The third dimension, activity understanding, focuses on the recognition of ongoing actions and interactions involving agents. This includes individual movements, co-presence of multiple agents, and interactions between agents and environmental elements. The evaluation prioritizes actions and interaction patterns that are central to scene dynamics, such as walking trajectories, social grouping, or engagement with shared spaces, rather than incidental or momentary motions. This dimension reflects the ability of the model to interpret dynamic processes that unfold within the scene.

The fourth dimension, holistic coherence, evaluates the overall integrative quality of the generated description. This dimension captures whether the model synthesizes individual observations into a coherent, plausible, and contextually grounded scene-level interpretation. High coherence implies internal consistency across object identification, spatial relations, and activity descriptions, as well as alignment with the functional and social context of the environment. Descriptions that are fragmented, internally contradictory, or contextually implausible receive lower evaluations under this criterion.

These four dimensions provide a structured yet flexible framework for assessing multimodal scene understanding in metaverse environments. They emphasize semantic fidelity and reasoning quality while accommodating the inherent openness of natural language descriptions.

6.3.2 Pairwise Comparison

Based on the defined semantic criteria, we adopt a pairwise comparison protocol implemented under an MLLM-as-a-judge paradigm. For each frame in the dataset, all candidate multimodal models are prompted with an identical instruction to generate a holistic, scene-level description. The resulting descriptions from two candidate models are then jointly presented to a separate judging model, GPT-5.2, which is instructed to compare their overall scene understanding quality and select the superior output based on the four semantic dimensions. The full judging prompt is illustrated in Fig. 6.1.

You are an expert evaluator for multimodal scene understanding in immersive metaverse environments.

You will be given:
1) An image representing a metaverse scene.
2) Two candidate scene descriptions generated by two different MLLMs (Model A and Model B).

Your task is to compare the two descriptions and determine which one demonstrates better overall scene understanding, based on the following semantic evaluation dimensions:

– **Object Recognition**: whether salient entities in the scene, including both static environmental elements and dynamic agents, are correctly and clearly identified.
– **Spatial Reasoning**: whether relative spatial relationships (e.g., proximity, orientation, containment) are accurately inferred and described.
– **Activity and Interaction Understanding**: whether ongoing actions, agent behaviors, and interactions among entities are correctly recognized.
– **Holistic Scene Coherence**: whether the description forms a consistent, complete, and plausible scene-level interpretation, rather than a collection of fragmented observations.

Please consider all four dimensions jointly and focus on the overall semantic quality of the scene understanding. Do not favor verbosity, stylistic preferences, or speculative details that are not grounded in the visual content.

Output only one of the following three options:
"Model A is better"
"Model B is better"
"Tie"

Do not provide explanations or additional commentary.

Figure 6.1: Prompt used for the MLLM-as-a-judge pairwise evaluation.

Pairwise comparison is chosen over absolute scoring for several reasons. First, open-ended scene descriptions do not admit a unique correct answer, rendering absolute scores difficult to calibrate across samples and evaluators. Second, absolute scoring schemes are prone to scale inconsistency and drift, particularly when judgments are aggregated across diverse scenes. In contrast, relative judgments focus on comparative semantic quality under identical visual inputs and task conditions, which improves robustness and reduces evaluator bias. Prior work has shown that such comparative evaluations yield more stable and reliable assessments for generative multimodal tasks [84].

In our evaluation protocol, LLaVA-v1.5-13B is selected as a fixed baseline model. All other multimodal models are compared both against this baseline and against one another across the dataset. This design anchors relative judgments and facilitates consistent cross-model comparison while still allowing the emergence of a global performance hierarchy.

6.3.3 Global Ranking with Elo Rating

While pairwise win rates provide an intuitive view of relative performance between individual model pairs, they do not directly produce a globally consistent ranking when multiple models are involved. To address this limitation, we adopt a bootstrap-based Elo rating framework following the methodology popularized by Chatbot Arena [85]. The Elo framework infers latent relative capability from repeated pairwise comparisons without requiring absolute ground-truth labels, making it well suited for open-ended generative evaluation.

In our setting, each model participates in 100 pairwise comparisons, ensuring sufficient comparison density for stable rating convergence. Let R_i and R_j denote the current Elo ratings of models i and j, respectively. The expected probability that model i is preferred over model j is defined as

$$E_i = \frac{1}{1 + 10^{(R_j - R_i)/400}}. \tag{6.1}$$

After each comparison, the rating of model i is updated according to

$$R_i' = R_i + K\left(S_i - E_i\right), \tag{6.2}$$

where $S_i \in \{0, 0.5, 1\}$ represents the observed outcome corresponding to a loss, tie, or win, and K controls the update step size.

By aggregating outcomes across many comparisons, the Elo framework produces a smooth and interpretable global ranking that reflects relative scene understanding capability across all evaluated models. Combined with the semantic evaluation dimensions and the pairwise judging protocol, this approach yields a principled and scalable evaluation methodology for benchmarking multimodal scene understanding in metaverse environments.

6.4 MLLM Evaluation Results

The pairwise comparison results for metaverse scene understanding are summarized in Table 6.1. The overall evaluation reveals a clear and stable performance stratification across the examined multimodal models, reflecting systematic differences in their ability to perceive, reason about, and coherently describe complex metaverse scenes. Both win rates and Elo scores exhibit consistent ordering, indicating that the adopted evaluation protocol yields robust and internally consistent outcomes.

At the top tier, closed-source frontier models demonstrate a pronounced advantage. Gemini-2.5-Pro achieves the strongest overall performance, with a win rate of 0.92 and the highest Elo score of 1420. This model consistently generates detailed and well-structured scene descriptions that accurately identify salient objects, maintain spatial consistency, and integrate agent activities into a coherent narrative. GPT-5-mini follows closely with a win rate of 0.86 and an Elo score of 1385, while Claude-Sonnet-4.5 attains a win rate of 0.84 and an Elo score of 1368. Across the dataset, these models exhibit superior robustness in handling visually dense scenes, multi-agent interactions, and diverse viewpoints. Their descriptions tend to balance

Table 6.1: Pairwise comparison results on metaverse scene understanding.

Model	Win / Tie / Lose	Win Rate	Elo Score
Gemini-2.5-Pro	368 / 25 / 7	0.92	1420
GPT-5-mini	344 / 15 / 41	0.86	1385
Claude-Sonnet-4.5	336 / 21 / 43	0.84	1368
Gemini-2.5-Flash	324 / 30 / 46	0.81	1342
LLaVA-v1.6-34B	312 / 35 / 53	0.78	1305
Qwen2-VL-72B	304 / 49 / 47	0.76	1287
LLaVA-v1.5-13B (anchor)	/	0.50	/
Gemini-1.0-Pro	176 / 51 / 173	0.44	1156
Qwen-VL-Chat	164 / 72 / 164	0.41	1232
Phi-3-Medium	152 / 77 / 171	0.38	1108
Yi-VL-6B	140 / 93 / 167	0.35	1085
MiniGPT-v2	124 / 94 / 182	0.31	1062
Cheetor	112 / 145 / 143	0.28	1095
LLaVA-v1.5-7B	104 / 118 / 178	0.26	1018

perceptual detail and semantic abstraction, avoiding both superficial enumeration and overgeneralized interpretations.

The strong performance of these frontier models suggests that large-scale multimodal pretraining and advanced cross-modal reasoning mechanisms play a critical role in metaverse scene understanding. In particular, their ability to integrate object recognition, spatial reasoning, and activity interpretation into a unified description contributes significantly to higher holistic coherence scores under the judging protocol.

Among open-source models, several high-capacity systems achieve competitive performance. LLaVA-v1.6-34B records a win rate of 0.78 and an Elo score of 1305,

while Qwen2-VL-72B achieves a win rate of 0.76 and an Elo score of 1287. Both models consistently outperform the anchor model, LLaVA-v1.5-13B, which is fixed at a win rate of 0.50. These results indicate that increased model scale and improved visual language alignment substantially enhance the ability to reason about complex spatial layouts and interaction patterns in metaverse environments. Although these models still lag behind closed-source frontier systems, the performance gap is narrower than that observed for smaller open-source alternatives, highlighting the effectiveness of recent advances in open multimodal modeling.

Models positioned below the anchor exhibit a noticeable decline in performance. Systems such as Gemini-1.0-Pro, Qwen-VL-Chat, and Phi-3-Medium show moderate win rates ranging from 0.38 to 0.44, suggesting partial competence in object recognition and basic spatial description, but reduced consistency in integrating these elements into a coherent scene-level interpretation. Lightweight or earlier-generation models, including Yi-VL-6B, MiniGPT-v2, Cheetor, and LLaVA-v1.5-7B, perform substantially worse, with win rates below 0.35. Qualitative inspection of their outputs reveals recurring issues such as fragmented descriptions, imprecise spatial relations, omission of key dynamic agents, and limited understanding of multi-agent interactions. These limitations become particularly pronounced in scenes with high visual complexity or concurrent activities.

Across all evaluated models, the Spearman rank correlation between win rates and Elo scores reaches $\rho = 0.97$, indicating a strong agreement between local pairwise comparison outcomes and the inferred global ranking. This high correlation confirms the internal consistency of the evaluation framework and supports the reliability of the derived performance hierarchy.

These results demonstrate that metaverse scene understanding remains a challenging task that strongly differentiates current multimodal models. While closed-source frontier models establish a clear upper bound in performance, recent large-scale open-source models show promising progress toward narrowing this gap. In contrast, lightweight models face substantial challenges in maintaining holistic coherence and

relational reasoning under complex, open-ended metaverse scenes. These findings underscore the importance of both model capacity and cross-modal reasoning design for advancing multimodal understanding in immersive virtual environments.

6.5 MLLM Benchmark Validity and Robustness Analysis

To examine the validity of the proposed multimodal large language model benchmarking framework, we conduct a human validation study that uses expert judgment as an external reference. Human evaluation is widely regarded as the most reliable standard for assessing open-ended scene understanding tasks, particularly in settings where no unique ground-truth description exists. By comparing automated judgments against expert consensus, this study evaluates whether the proposed framework yields assessments that are both meaningful and aligned with human perception.

6.5.1 Human Validation and Agreement Analysis

Three human experts with backgrounds in immersive systems, virtual environments, and multimodal perception are invited to participate in the validation process. Each expert independently evaluates a randomly sampled subset of 100 metaverse frames drawn from the benchmark dataset. For each frame, experts are provided with identical visual inputs and task instructions, mirroring the conditions under which the multimodal models and the automated judge operate. The experts assess pairwise model outputs by selecting the description that demonstrates superior scene understanding based on their professional judgment.

To reduce individual subjectivity, expert judgments are aggregated using majority voting to form a human consensus for each comparison. This consensus serves as a reference point for evaluating the reliability of automated judgments. We then compare the human consensus against judgments produced by GPT-5.2 under two

distinct evaluation settings. In the first setting, GPT-5.2 operates under explicit guidance from the proposed metaverse-specific semantic evaluation criteria, including object recognition, spatial reasoning, activity understanding, and holistic coherence. In the second setting, GPT-5.2 performs comparisons without access to these criteria and relies solely on its general preference when selecting the superior description.

The results reveal a substantial difference between the two settings. When guided by the proposed criteria, GPT-5.2 achieves an agreement rate of 87.6% with human consensus. In contrast, when the criteria are removed, the agreement rate decreases to 78.9%. This improvement demonstrates that the proposed criteria significantly enhance the alignment between automated judgments and expert human evaluations. The high agreement rate under the structured setting indicates that the judge model, when properly guided, attains near-human reliability in comparative metaverse scene understanding. More importantly, the observed performance gap highlights the role of the criteria as an effective inductive structure, directing the judge to attend to semantically salient aspects of immersive scenes rather than relying on superficial or stylistic cues.

6.5.2 Robustness to Positional Bias

Beyond overall effectiveness, the robustness of the benchmarking framework is further examined by explicitly addressing common sources of evaluation bias. The first and most fundamental concern is positional bias, which refers to a systematic tendency of an evaluator to favor responses based on their presentation order rather than their intrinsic quality. In pairwise comparison settings, such bias may arise when the first response benefits from primacy effects or when later responses suffer from reduced attention. For example, if two answers of comparable quality are presented sequentially, an evaluator may consistently prefer the first answer simply because it establishes an initial reference point, even when the second answer provides more accurate or detailed information. In this case, the evaluation outcome reflects presentation effects

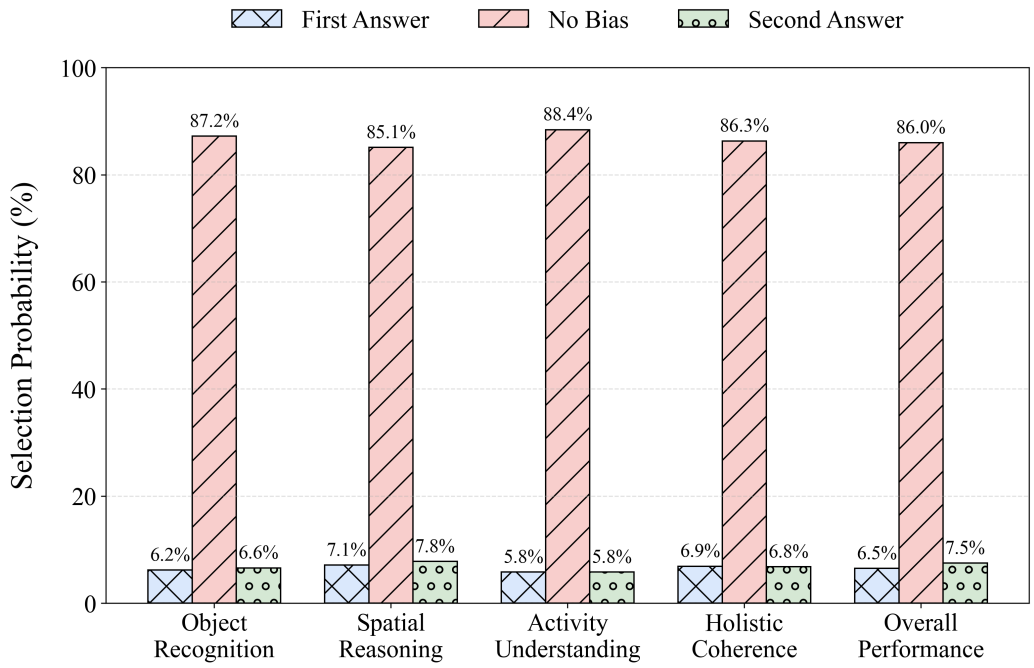
rather than genuine performance differences. If left unexamined, positional bias can severely undermine the credibility of pairwise benchmarking results, as model rankings may be driven by ordering artifacts instead of true semantic understanding or reasoning ability.

To assess positional bias, we conduct paired evaluations in which, for each image and model pair, the two candidate descriptions are presented in reversed order. The judge model evaluates both orderings independently. Positional bias is analyzed separately for each semantic evaluation dimension, including object recognition, spatial reasoning, activity understanding, and holistic scene coherence, as well as for the aggregated judgment across all dimensions.

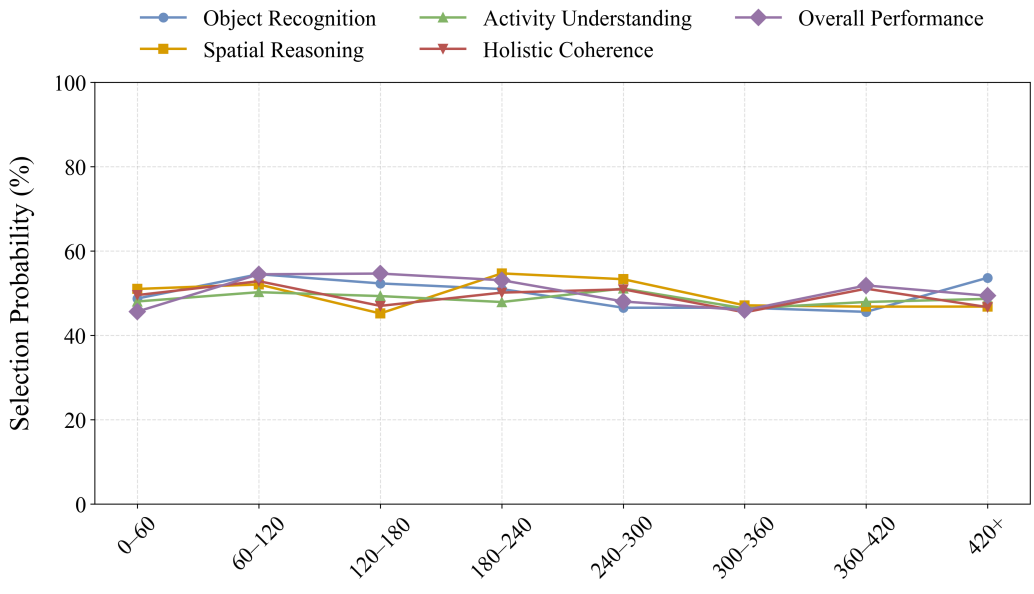
As illustrated in Fig. 6.2(a), the majority of judgments remain invariant to the presentation order. Across all semantic dimensions and in the overall analysis, more than 85% of comparisons yield identical outcomes after reordering. The remaining cases are divided between preferences for the first and second positions, with both proportions remaining low and closely balanced. The absence of a dominant positional preference indicates that the judge model's decisions are primarily driven by semantic quality rather than answer order. These findings suggest that the proposed evaluation framework exhibits strong robustness to positional bias across both individual dimensions and holistic judgments.

6.5.3 Robustness to Length Bias

We further investigate length bias, which refers to a systematic tendency of the judge to favor longer or more verbose responses regardless of their actual semantic accuracy, relevance, or usefulness. This form of bias is particularly prevalent in generative evaluation settings, where verbosity may be mistakenly interpreted as greater informativeness or reasoning depth. For instance, when comparing two responses to the same question, a shorter answer may deliver a precise and correct explanation, while a longer answer elaborates extensively but includes redundant, vague, or even partially



(a) Position bias.



(b) Length bias.

Figure 6.2: Bias analysis in MLLM-based pairwise evaluation: (a) positional bias under response order permutation; (b) length bias across response length differences.

incorrect content. If the judge consistently prefers the latter simply due to its length, the evaluation outcome becomes skewed toward verbosity rather than quality. Such bias can inflate the perceived performance of models that generate longer outputs, while undervaluing concise models that produce accurate and well-focused responses, thereby distorting comparative benchmarking results.

For each pairwise comparison, we compute the difference in response length between the two candidate descriptions and group comparisons into predefined length-difference bins. For each bin, we analyze the probability that the longer response is selected as superior. This analysis is conducted separately for each semantic evaluation dimension and for the overall judgment, as shown in Fig. 6.2(b).

The results indicate no systematic preference for longer or shorter responses. Across all length-difference bins, the selection probability remains centered around 0.5 and is confined within a narrow range of approximately 0.45 to 0.55. Moreover, no monotonic trend is observed as the length difference increases. These patterns are consistent across individual semantic dimensions and the aggregated evaluation. The absence of length-dependent preference demonstrates that the judge model does not equate verbosity with quality and instead bases its decisions on semantic relevance, correctness, and coherence.

Overall, the human validation and bias analyses jointly demonstrate that the proposed benchmarking framework is both effective and robust. The close alignment with expert judgments, combined with strong resistance to positional and length biases, provides empirical support for the reliability of the framework as a tool for evaluating multimodal scene understanding in metaverse environments.

6.6 Limitations

Although this chapter proposes a multimodal benchmarking framework for evaluating metaverse scene understanding, several limitations should be acknowledged.

First, the constructed dataset is limited in scale, source diversity, and scene coverage. The benchmark contains 400 frames sampled from representative metaverse environments, including platforms such as Roblox, Minecraft, and VRChat. These frames cover diverse visual elements, spatial layouts, viewpoints, and avatar interactions, providing a useful foundation for evaluating metaverse-oriented scene understanding. However, the dataset is still relatively small compared with large-scale multimodal benchmarks. It may not fully capture the diversity of metaverse environments, such as highly customized virtual worlds, industrial digital twins, educational simulations, collaborative design spaces, and large-scale social events. In addition, the current benchmark is mainly based on image-level scene understanding. Although the frames are sampled from continuous rendering sequences, the evaluation does not explicitly model long-term temporal dynamics, continuous user behavior, or evolving multi-agent interactions. Therefore, future work should expand the dataset with more platforms, scene types, interaction patterns, and video-based samples to better reflect the complexity of real metaverse applications.

Second, the evaluation is based on a selected group of representative MLLMs and a controlled prompting protocol, which may limit the generality of the model comparison. The benchmark includes closed-source frontier models, large-scale open-source models, and lightweight open-source models, allowing a structured comparison across different model categories. However, the selected models cannot exhaustively represent the rapidly evolving landscape of multimodal large language models. Model capabilities, API behaviors, and decoding strategies may change over time, especially for closed-source systems. In addition, although the same prompt is used across models to ensure fairness, model outputs may still be sensitive to prompt wording, system instructions, image preprocessing, and decoding configurations. Some models may benefit more from task-specific prompting or multi-turn reasoning, while others may perform better under concise instructions. Therefore, the reported rankings should be interpreted as the results under the specific model versions and evaluation settings used in this chapter, rather than as permanent or universal conclusions about

all MLLMs.

Third, the proposed MLLM-as-a-judge protocol improves scalability, but it still depends on automated judgment and limited human validation. Pairwise comparison provides a practical way to evaluate open-ended scene descriptions without requiring a single ground-truth reference. The use of metaverse-specific semantic criteria also improves the alignment between GPT-based judgments and human expert consensus. Nevertheless, the judge model may still contain implicit biases or reasoning limitations that are not fully captured by the positional and length bias analyses. For example, it may favor descriptions that appear semantically rich but contain subtle visual inaccuracies, or it may overlook fine-grained spatial errors in complex scenes. Moreover, the human validation study is conducted with three experts on a subset of 100 frames, which provides useful evidence of reliability but may not fully represent broader user judgments across different backgrounds and task contexts. Future work should involve larger-scale human evaluation, multiple judge models, additional bias tests, and task-based validation to further strengthen the reliability and practical relevance of the benchmarking framework.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This dissertation explores how agentic AI can be systematically embedded into metaverse systems in order to overcome the inherent limitations of existing virtual platforms in intelligence, adaptability, and real-time responsiveness. Traditional metaverse systems largely rely on pre-defined logic, reactive control, and centralized resource management, which constrains their ability to support persistent perception, dynamic coordination, and scalable user interaction. By reframing the metaverse as an agentic system composed of heterogeneous, goal-oriented AI agents, this work provides a unified perspective that integrates architectural design, adaptive resource allocation, and multimodal cognitive evaluation within a single coherent framework.

At the system level, this thesis proposes a unified thing–edge–cloud architecture that organizes heterogeneous agents according to their computational characteristics, latency sensitivity, and functional roles. Rather than treating devices, edge servers, and cloud platforms as isolated components, the proposed architecture enables coordinated perception, reasoning, and action across layers. Lightweight agents deployed at the thing and edge layers are responsible for real-time interaction, environment sensing, and latency-critical decision-making, while cloud-level agents provide global reasoning, long-horizon planning, and high-level cognitive support. Central to this architecture is an AI agent controller that performs task cognition, task decompo-

sition, cross-layer allocation, and performance-aware re-planning. By dynamically mapping subtasks to appropriate agents and execution layers, the controller serves as the core orchestration mechanism that enables adaptive and scalable agentic behavior in complex metaverse environments.

To quantitatively evaluate user experience under such a dynamic and distributed system, this work further introduces a time-step-based QoE model specifically tailored to metaverse applications. Unlike traditional QoE models that rely on coarse-grained or static indicators, the proposed model captures fine-grained temporal dynamics by incorporating real-time measures of scene quality, frame continuity, and latency variation. These indicators are designed to reflect the perceptual sensitivity of immersive users and the interactive nature of metaverse experiences. The proposed QoE model is rigorously validated through controlled subjective Mean Opinion Score experiments, and the results demonstrate a strong correlation between the modeled QoE and user-perceived quality. This validation confirms the suitability of the model as a reliable feedback signal for adaptive control and learning-based optimization in agentic metaverse systems.

Building on the proposed QoE formulation, this thesis develops a novel adaptive resource allocation algorithm based on SAC algorithm. To explicitly capture the interaction structure among concurrent users, the algorithm models users as agents embedded in a dynamic graph, where edges represent network contention and shared resource dependencies. Graph convolutional networks are employed to extract relational features, while self-attention mechanisms enable agents to selectively focus on the most relevant neighbors under varying network conditions. This design supports decentralized yet coordinated decision-making, allowing each agent to adapt its allocation strategy while implicitly accounting for the global system state. Extensive experimental results demonstrate that the proposed SAC-GCN approach consistently outperforms traditional congestion control methods as well as baseline deep reinforcement learning algorithms across diverse network scenarios. The gains are reflected not only in higher overall QoE, but also in improved resource utilization efficiency

and a more balanced allocation of communication and computation resources.

Beyond system architecture and resource optimization, this dissertation addresses the challenge of evaluating cognitive perception in metaverse environments. A metaverse-specific scene understanding dataset is constructed based on a high-fidelity Unreal Engine environment that captures realistic spatial layouts, dynamic agents, and interaction patterns. To evaluate open-ended scene understanding, this work introduces an MLLM-as-a-judge benchmarking paradigm guided by semantic criteria tailored to metaverse perception, including object recognition, spatial reasoning, activity understanding, and holistic coherence. Through extensive pairwise comparisons and Elo-based ranking, the benchmark provides a robust and scalable method for assessing multimodal scene understanding without relying on rigid ground-truth annotations. Human validation and bias analysis further demonstrate that the proposed evaluation framework achieves near-human reliability while remaining robust to common sources of evaluation bias.

This dissertation advances the state of the art in metaverse research by establishing an integrated agentic framework that spans system architecture, adaptive resource allocation, and cognitive benchmarking. By combining agentic AI principles with rigorous modeling, learning-based optimization, and structured evaluation, the work provides both theoretical insights and practical methodologies for building intelligent, adaptive, and user-centric metaverse systems. The proposed approaches lay a foundation for future research on large-scale agent coordination, edge-native intelligence, and multimodal cognition in next-generation immersive environments.

7.2 Future Work

While this dissertation establishes a comprehensive foundation for agentic metaverse systems, it also opens multiple avenues for future investigation that extend beyond the current scope. These directions span agent cognition, system architecture, evaluation methodology, and real-world deployment considerations, collectively pointing toward

more autonomous, scalable, and socially embedded metaverse systems.

A first promising direction concerns the development of richer agent memory mechanisms and long-term learning capabilities. The agents considered in this work primarily operate under episodic decision horizons and rely on short-term observations or learned policies. Future research could integrate structured memory modules, such as episodic memory, semantic memory, or world models, enabling agents to accumulate experience across extended user sessions. Such memory-enhanced agents could support continual learning, adapt to long-term user preferences, and respond more effectively to evolving environments. This direction is particularly relevant for persistent metaverse worlds, where historical context and long-term consistency play a critical role in user immersion and system intelligence.

A second direction involves deeper integration between cognitive reasoning and physical simulation. Although current agentic frameworks enable perception-driven decision-making, reasoning is often decoupled from the underlying physics and environment dynamics. Future systems could more tightly couple high-level reasoning processes with real-time physical simulation, allowing agents to anticipate environmental changes, predict user behavior, and proactively adapt the virtual world. This tighter coupling may enable more advanced forms of embodied intelligence, where agents not only react to the environment but also actively reshape it to optimize interaction quality, safety, or efficiency.

Third, the evaluation framework introduced in this dissertation can be extended along several dimensions to better reflect realistic metaverse usage. One important extension is toward multi-turn and interactive evaluation, where models are assessed based on their ability to maintain consistency, context awareness, and goal alignment across sequential interactions rather than isolated frames. Additionally, incorporating multi-user collaboration scenarios would enable the evaluation of social reasoning, coordination, and conflict resolution among agents and users. Real-time decision feedback, where agent actions influence subsequent observations and evaluations, would further bridge the gap between offline benchmarking and online deployment.

Another important research direction relates to the scalability and robustness of agentic metaverse systems under heterogeneous and large-scale deployment conditions. Future work could investigate hierarchical or modular agent organizations that support thousands of concurrent users while maintaining coordination efficiency. Adaptive mechanisms for handling network volatility, device heterogeneity, and sudden workload spikes will be essential to ensure stable performance in real-world environments. Integrating predictive models for user behavior and network dynamics may further enhance system resilience and responsiveness.

From a practical deployment perspective, privacy, energy efficiency, and sustainability represent critical challenges. Future agentic metaverse architectures should incorporate privacy-aware perception and decision-making mechanisms, minimizing unnecessary data transmission and enabling user-controllable data sharing policies. Energy-efficient agent design, particularly for edge and device-level agents, will be essential to reduce operational costs and environmental impact. Federated and decentralized learning paradigms offer a promising pathway to address these concerns by enabling collaborative model improvement without centralized data aggregation, while also aligning with emerging regulatory and ethical requirements.

Finally, interdisciplinary exploration represents a valuable avenue for future research. Combining insights from human–computer interaction, cognitive science, social computing, and ethics could lead to more human-centered agentic metaverse systems. Understanding how users perceive, trust, and collaborate with autonomous agents will be crucial for designing systems that are not only technically effective but also socially acceptable and inclusive.

References

[1] Neal Stephenson. *Snow crash*. Penguin UK, 1994.

[2] Maged N Kamel Boulos, Lee Hetherington, and Steve Wheeler. Second life: an overview of the potential of 3-d virtual worlds in medical and health education. *Health Information & Libraries Journal*, 24(4):233–245, 2007.

[3] Yuntao Wang, Zhou Su, Ning Zhang, Rui Xing, Dongxiao Liu, Tom H Luan, and Xuemin Shen. A survey on metaverse: Fundamentals, security, and privacy. *IEEE Communications Surveys & Tutorials*, 2022.

[4] Yogesh K Dwivedi, Laurie Hughes, Abdullah M Baabdullah, Samuel Ribeiro-Navarrete, Mihalis Giannakis, Mutaz M Al-Debei, Denis Dennehy, Bhimaraya Metri, Dimitrios Buhalis, Christy MK Cheung, et al. Metaverse beyond the hype: Multidisciplinary perspectives on emerging challenges, opportunities, and agenda for research, practice and policy. *International journal of information management*, 66:102542, 2022.

[5] Jackie Wiles. What is a Metaverse? And should you be buying in? https://www.gartner.com/en/articles/what-is-a-metaverse, 2022.

[6] John David N Dionisio, William G Burns Iii, and Richard Gilbert. 3d virtual worlds and the metaverse: Current status and future possibilities. *ACM computing surveys (CSUR)*, 45(3):1–38, 2013.

[7] Hang Wang, Huansheng Ning, Yujia Lin, Wenxi Wang, Sahraoui Dhelim, Fadi Farha, Jianguo Ding, and Mahmoud Daneshmand. A survey on the metaverse: The state-of-the-art, technologies, applications, and challenges. *IEEE Internet of Things Journal*, 10(16):14671–14688, 2023.

[8] Abdulmotaleb El Saddik. Digital twins: The convergence of multimedia technologies. *IEEE Multimedia*, 25(2):87–92, 2018.

[9] Wei Yang, Wei Xiang, Yuan Yang, and Peng Cheng. Optimizing federated learning with deep reinforcement learning for digital twin empowered industrial IoT. *IEEE Transactions on Industrial Informatics*, 19(2):1884–1893, 2022.

[10] Mahshid Mehrabi, Dongho You, Vincent Latzko, Hani Salah, Martin Reisslein, and Frank HP Fitzek. Device-enhanced mec: Multi-access edge computing (mec) aided by end device computation and caching: A survey. *IEEE Access*, 7:166079–166108, 2019.

[11] Dario Sabella, Alessandro Vaillant, Pekka Kuure, Uwe Rauschenbach, and Fabio Giust. Mobile-edge computing architecture: The role of mec in the internet of things. *IEEE Consumer Electronics Magazine*, 5(4):84–91, 2016.

[12] Zijian Long, Haiwei Dong, and Abdulmotaleb El Saddik. Human-centric resource allocation for the metaverse with multi-access edge computing. *IEEE Internet of Things Journal*, 10(22):19993–20005, 2023.

[13] Fengxiao Tang, Xuehan Chen, Ming Zhao, and Nei Kato. The roadmap of communication and networking in 6G for the metaverse. *IEEE Wireless Communications*, 2022.

[14] Luyi Chang, Zhe Zhang, Pei Li, Shan Xi, Wei Guo, Yukang Shen, Zehui Xiong, Jiawen Kang, Dusit Niyato, Xiuquan Qiao, et al. 6g-enabled edge ai for metaverse: Challenges, methods, and future research directions. *Journal of communications and information networks*, 7(2):107–121, 2022.

[15] Mona M Soliman, Eman Ahmed, Ashraf Darwish, and Aboul Ella Hassanien. Artificial intelligence powered metaverse: analysis, challenges and future perspectives. *Artificial Intelligence Review*, 57(2):36, 2024.

[16] Thien Huynh-The, Quoc-Viet Pham, Xuan-Qui Pham, Thanh Thi Nguyen, Zhu Han, and Dong-Seong Kim. Artificial intelligence for the metaverse: A survey. *Engineering Applications of Artificial Intelligence*, 117:105581, 2023.

[17] Thippa Reddy Gadekallu, Thien Huynh-The, Weizheng Wang, Gokul Yenduri, Pasika Ranaweera, Quoc-Viet Pham, Daniel Benevides da Costa, and Madhusanka Liyanage. Blockchain for the metaverse: A review. *arXiv preprint arXiv:2203.09738*, 2022.

[18] Thien Huynh-The, Thippa Reddy Gadekallu, Weizheng Wang, Gokul Yenduri, Pasika Ranaweera, Quoc-Viet Pham, Daniel Benevides da Costa, and Madhusanka Liyanage. Blockchain for the metaverse: A review. *Future Generation Computer Systems*, 143:401–419, 2023.

[19] Aishik Ghosh, Vikas Hassija, Vinay Chamola, Abdulmotaleb El Saddik, et al. A survey on decentralized metaverse using blockchain and web 3.0 technologies, applications, and more. *IEEE Access*, 2024.

[20] Erqiang Deng, Li You, Fazlullah Khan, Guosong Zhu, Zhen Qin, Saru Kumari, Hu Xiong, and Ryan Alturki. Decoding the third dimension in the metaverse: A comprehensive method for reconstructing 2d nft portraits into 3d models. *Applied Soft Computing*, 165:111964, 2024.

[21] Åsa Fast-Berglund, Liang Gong, and Dan Li. Testing and validating extended reality (xR) technologies in manufacturing. *Procedia Manufacturing*, 25:31–38, 2018.

[22] Paul Milgram and Fumio Kishino. A taxonomy of mixed reality visual displays. *IEICE Transactions on Information and Systems*, 77(12):1321–1329, 1994.

[23] Philip Rosedale. Virtual reality: The next disruptor: A new kind of worldwide communication. *IEEE Consumer Electronics Magazine*, 6(1):48–50, 2017.

[24] Yu Yuan. Paving the road for virtual and augmented reality [Standards]. *IEEE Consumer Electronics Magazine*, 7(1):117–128, 2018.

[25] Longyu Zhang, Sifeng Chen, Haiwei Dong, and Abdulmotaleb El Saddik. Visualizing Toronto city data with HoloLens: Using augmented reality for a city model. *IEEE Consumer Electronics Magazine*, 7(3):73–80, 2018.

[26] Jing Song, Ya Kang, Qingyang Song, Lei Guo, and Abbas Jamalipour. Distributed resource optimization with blockchain security for immersive digital twin in IIoT. *IEEE Transactions on Industrial Informatics*, 19(5):7258–7267, 2022.

[27] Jiacheng Xie, Yali Liu, Xuewen Wang, Shukai Fang, and Shuguang Liu. A new xr-based human-robot collaboration assembly system based on industrial metaverse. *Journal of Manufacturing Systems*, 74:949–964, 2024.

[28] Deepak Bhaskar Acharya, Karthigeyan Kuppan, and B. Divya. Agentic AI: Autonomous intelligence for complex goals—A comprehensive survey. *IEEE Access*, 2025.

[29] Feibo Jiang, Cunhua Pan, Li Dong, Kezhi Wang, Octavia A. Dobre, and Merouane Debbah. From large AI models to agentic AI: A tutorial on future intelligent communications. *arXiv preprint arXiv:2505.22311*, 2025.

[30] Ranjan Sapkota, Konstantinos I. Roumeliotis, and Manoj Karkee. AI agents vs. agentic AI: A conceptual taxonomy, applications, and challenges. *Information Fusion*, 126:103599, 2026.

[31] Shujia Fan, Brian Yecies, Zeyang Ivy Zhou, and Jun Shen. Challenges and opportunities for the web 3.0 metaverse turn in education. *IEEE Transactions on Learning Technologies*, 17:1935–1950, 2024.

[32] Herman Zahid, Adil Zulfiqar, Muhammad Adnan, Muhammad Sajid Iqbal, Anwar Shah, Usman Abbasi, and Salah Eldeen Gasim Mohamed. Transforming

nano grids to smart grid 3.0: Ai, digital twins, blockchain, and the metaverse revolutionizing the energy ecosystem. *Results in Engineering*, page 105850, 2025.

[33] Andreas M Kaplan and Michael Haenlein. The fairyland of second life: Virtual social worlds and how to use them. *Business horizons*, 52(6):563–572, 2009.

[34] Rosa Mikeal Martey and Jennifer Stromer-Galley. The digital dollhouse: Context and social norms in The Sims Online. *Games and Culture*, 2(4):314–334, 2007.

[35] Mark G Chen. Communication, coordination, and camaraderie in World of Warcraft. *Games and Culture*, 4(1):47–73, 2009.

[36] Omar Ali, Mujtaba Momin, Anup Shrestha, Ronnie Das, Fadia Alhajj, and Yogesh K Dwivedi. A review of the key challenges of non-fungible tokens. *Technological Forecasting and Social Change*, 187:122248, 2023.

[37] Marcus Carter, Kyle Moore, Jane Mavoa, Heather Horst, and Luke Gaspard. Situating the appeal of Fortnite within children's changing play cultures. *Games and Culture*, 15(4):453–471, 2020.

[38] Barbara Guidi and Andrea Michienzi. Social games and Blockchain: Exploring the metaverse of Decentraland. In *2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops*, pages 199–204. IEEE, 2022.

[39] Benjamin Rainer, Stefan Petscharnig, Christian Timmerer, and Hermann Hellwagner. Statistically indifferent quality variation: An approach for reducing multimedia distribution cost for adaptive video streaming services. *IEEE Transactions on Multimedia*, 19(4):849–860, 2016.

[40] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. Neural adaptive video streaming with pensieve. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 197–210, 2017.

[41] Tianchi Huang, Xin Yao, Chenglei Wu, Rui-Xiao Zhang, Zhengyuan Pang, and Lifeng Sun. Tiyuntsong: A self-play reinforcement learning approach for ABR

video streaming. In *Proceedings of IEEE International Conference on Multimedia and Expo*, pages 1678–1683, 2019.

[42] Hao Chen, Xu Zhang, Yiling Xu, Ju Ren, Jingtao Fan, Zhan Ma, and Wenjun Zhang. T-gaming: A cost-efficient cloud gaming system at scale. *IEEE Transactions on Parallel and Distributed Systems*, 30(12):2849–2865, 2019.

[43] Zhaocheng Wang, Jun Wu, Rui Wang, and Ying Li. Distributed multi-agent resource allocation based on transformer and drl for cloud xr hybrid content transmission. *IEEE Internet of Things Journal*, 2025.

[44] Haopeng Wang, Zijian Long, Haiwei Dong, and Abdulmotaleb El Saddik. Madrl-based rate adaptation for 360° video streaming with multiviewpoint prediction. *IEEE Internet of Things Journal*, 11(15):26503–26517, 2024.

[45] Fazal E Subhan, Abid Yaqoob, Cristina Hava Muntean, and Gabriel-Miro Muntean. A survey on artificial intelligence techniques for improved rich media content delivery in a 5g and beyond network slicing context. *IEEE Communications Surveys & Tutorials*, 2024.

[46] Wenchao Liu, Hao Wang, Xuhui Zhang, Huijun Xing, Jinke Ren, Yanyan Shen, and Shuguang Cui. Joint trajectory design and resource allocation in uav-enabled heterogeneous mec systems. *IEEE Internet of Things Journal*, 2024.

[47] Mohsen Khani, Mohammad Mohsen Sadr, and Shahram Jamali. Deep reinforcement learning-based resource allocation in multi-access edge computing. *Concurrency and Computation: Practice and Experience*, 36(15):e7995, 2024.

[48] Caio Souza, Marcos Falcão, Andson Balieiro, Elton Alves, and Tarik Taleb. Dynamic resource allocation for urllc and embb in mec-nfv 5g networks. *Computer Networks*, 260:111127, 2025.

[49] Zhen Wang, Bin Lin, Qiang Ye, Yuguang Fang, and Xiaoling Han. Joint computation offloading and resource allocation for maritime mec with energy harvesting. *IEEE Internet of Things Journal*, 11(11):19898–19913, 2024.

[50] Jiadai Wang, Lei Zhao, Jiajia Liu, and Nei Kato. Smart resource allocation for mobile edge computing: A deep reinforcement learning approach. *IEEE Transactions on Emerging Topics in Computing*, 9(3):1529–1541, 2019.

[51] Xiaolan Liu, Jiadong Yu, Zhiyong Feng, and Yue Gao. Multi-agent reinforcement learning for resource allocation in IoT networks with edge computing. *China Communications*, 17(9):220–236, 2020.

[52] Muhammet Hevesli, Abegaz Mohammed Seid, Aiman Erbad, and Mohamed Abdallah. Multi-agent drl for queue-aware task offloading in hierarchical mec-enabled air-ground networks. *IEEE Transactions on Cognitive Communications and Networking*, 2025.

[53] Nam H Chu, Hieu Q Nguyen, Diep N Nguyen, Dinh Thai Hoang, Khoa T Phan, Eryk Dutkiewicz, Dusit Niyato, and Tao Shu. Dynamic multi-tier resource allocation framework for metaverse. *IEEE Network*, 2024.

[54] Jiadong Yu, Ahmad Yousef Alhilal, Tailin Zhou, Pan Hui, and Danny HK Tsang. Attention-based qoe-aware digital twin empowered edge computing for immersive virtual reality. *IEEE Transactions on Wireless Communications*, 23(9):11276–11290, 2024.

[55] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems*, 30:6382–6393, 2017.

[56] Jiechuan Jiang, Chen Dun, Tiejun Huang, and Zongqing Lu. Graph convolutional reinforcement learning. *arXiv preprint arXiv:1810.09202*, 2018.

[57] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. Mean field multi-agent reinforcement learning. In *Proceedings of International Conference on Machine Learning*, pages 5571–5580, 2018.

[58] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[59] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zi-Hang Jiang, Francis E. H. Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token ViT: Training vision transformers from scratch on ImageNet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 558–567, 2021.

[60] Duo Zheng, Shijia Huang, and Liwei Wang. Video-3D LLM: Learning position-aware video representation for 3D scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8995–9006, 2025.

[61] Weizhen Wang, Chenda Duan, Zhenghao Peng, Yuxin Liu, and Bolei Zhou. Embodied scene understanding for vision-language models via MetaVQA. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22453–22464, 2025.

[62] Dongping Chen, Ruoxi Chen, Shilin Zhang, Yaochen Wang, Yinuo Liu, Huichi Zhou, Qihui Zhang, Yao Wan, Pan Zhou, and Lichao Sun. MLLM-as-a-judge: Assessing multimodal LLM-as-a-judge with vision-language benchmarks. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[63] Yang Liu, Haiwei Dong, Longyu Zhang, and Abdulmotaleb El Saddik. Technical evaluation of HoloLens for multimedia: A first look. *IEEE MultiMedia*, 25(4):8–18, 2018.

[64] Longyu Zhang, Haiwei Dong, and Abdulmotaleb El Saddik. Towards a QoE model to evaluate holographic augmented reality devices. *IEEE MultiMedia*, 26(2):21–32, 2019.

[65] ArcGIS Living Atlas of the World. https://livingatlas.arcgis.com/.

[66] Markus Funk, Mareike Kritzler, and Florian Michahelles. HoloLens is more than air tap: Natural and intuitive interaction with holograms. In *Proceedings of the Seventh International Conference on the Internet of Things*, pages 1–2, 2017.

[67] Luyang Liu, Ruiguang Zhong, Wuyang Zhang, Yunxin Liu, Jiansong Zhang, Lintao Zhang, and Marco Gruteser. Cutting the cord: Designing a high-quality untethered VR system with low latency remote rendering. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, pages 68–80, 2018.

[68] Mattia Lecci, Matteo Drago, Andrea Zanella, and Michele Zorzi. An open framework for analyzing and modeling XR network traffic. *IEEE Access*, 9:129782–129795, 2021.

[69] Savera Tanwir and Harry Perros. A survey of VBR video traffic models. *IEEE Communications Surveys Tutorials*, 15(4):1778–1802, 2013.

[70] Branislav Sredojev, Dragan Samardzija, and Dragan Posarac. WebRTC technology overview and signaling solution design and implementation. In *Proceedings of the 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1006–1009, 2015.

[71] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, and Janardhan Iyengar. The QUIC transport protocol: Design and internet-scale deployment. In *Proceedings of the ACM Special Interest Group on Data Communication*, pages 183–196, 2017.

[72] Adeel Ehsan, Mohammed Ahmad M. E. Abuhaliqa, Cagatay Catal, and Deepti Mishra. RESTful API testing methodologies: Rationale, challenges, and solution directions. *Applied Sciences*, 12(9):4369, 2022.

[73] Andrea Sgambelluri, Alessandro Pacini, Francesco Paolucci, Piero Castoldi, and Luca Valcarenghi. Reliable and scalable Kafka-based framework for optical network telemetry. *Journal of Optical Communications and Networking*, 13(10):E42–E52, 2021.

[74] Philippe Dobbelaere and Kyumars Sheykh Esmaili. Kafka versus RabbitMQ: A comparative study of two industry reference publish/subscribe implementations: Industry paper. In *Proceedings of the 11th ACM International Conference on Distributed and Event-Based Systems*, pages 227–238, 2017.

[75] Abul Ehtesham, Aditi Singh, Gaurav Kumar Gupta, and Saket Kumar. A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A), and Agent Network Protocol (ANP). *arXiv preprint arXiv:2505.02279*, 2025.

[76] Xinyi Hou, Yanjie Zhao, Shenao Wang, and Haoyu Wang. Model Context Protocol (MCP): Landscape, security threats, and future research directions. *arXiv preprint arXiv:2503.23278*, 2025.

[77] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

[78] Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-Draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663–704, 2008.

[79] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic

actor. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1861–1870, 2018.

[80] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-learning. In *Proceedings of AAAI Conference on Artificial Intelligence*, page 2094–2100, 2016.

[81] Jukka K Nurminen, Antony JR Meyn, Eetu Jalonen, Yrjo Raivio, and Raúl Garcıa Marrero. P2P media streaming with HTML5 and WebRTC. In *Proceedings of Conference on Computer Communications Workshops*, pages 63–64, 2013.

[82] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[83] Gaetano Carlucci, Luca De Cicco, Stefan Holmer, and Saverio Mascolo. Analysis and design of the Google congestion control for web real-time communication. In *Proceedings of the 7th International Conference on Multimedia Systems*, pages 1–12, 2016.

[84] Wentao Ge, Shunian Chen, Hardy Chen, Nuo Chen, Junying Chen, Zhihong Chen, Wenya Xie, Shuo Yan, ChenghaoZhu ChenghaoZhu, Ziyue Lin, Dingjie Song, Xidong Wang, Anningzhe Gao, Zhang Zhiyi, Jianquan Li, Xiang Wan, and Benyou Wang. MLLM-bench: Evaluating multimodal LLMs with per-sample criteria. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics*, pages 4951–4974, 2025.

[85] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and

Chatbot Arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.