A Study on Delay and Traffic Reduction for Cloud Gaming Systems

Submitted to
Graduate School of Information Science and Technology
Osaka University

January 2024

Takumasa ISHIOKA

List of Publications

Related Journal Articles

1. Takumasa Ishioka, Tatsuya Fukui, Ryouhei Tsugami, Toshihito Fujiwara, Satoshi Narikawa, Takuya Fujihashi, Shunsuke Saruwatari, Takashi Watanabe, "Traffic Reduction for Speculative Video Transmission in Cloud Gaming Systems Volume and Number", *IEICE Transactions On Communication*, Vol.E107-B, No.5, pp. 1–11, May. 2024, (in press).

2. Takumasa Ishioka, Tatsuya Fukui, Ryouhei Tsugami, Toshihito Fujiwara, Satoshi Narikawa, Takuya Fujihashi, Shunsuke Saruwatari, Takashi Watanabe, "Pattern Reduction for Low-Traffic Speculative Video Transmission in Cloud Gaming System", *IEEE Access*, (accepted).

3. Takumasa Ishioka, Kazuki Aiura, Ryota Shiina, Tatsuya Fukui, Tomohiro Taniguchi, Satochi Narikawa, Katsuya Minami, Kazuhiro Kizaki, Takuya Fujihashi, Takashi Watanabe, Shunsuke Saruwatari, "Design and Prototype Implementation of Software-Defined Radio Over Fiber", *IEEE Access*, Vol. 9, pp.72793–72807, 2021.

Related Conference Papers

1. Takumasa Ishioka, Tatsuya Fukui, Ryouhei Tsugami, Toshihito Fujiwara, Satoshi Narikawa, Takuya Fujihashi, Shunsuke Saruwatari, Takashi Watanabe, "Improving Quality of Experience in Cloud Gaming Using Speculative Execution", The 14th International Conference on Mobile Computing and Ubiquitous Networking (IEEE ICMU'23), pp.1–4, November 2023.

2. Takumasa Ishioka, Tatsuya Fukui, Ryouhei Tsugami, Toshihito Fujiwara, Satoshi Narikawa, Takuya Fujihashi, Shunsuke Saruwatari, Takashi Watanabe, "A Concept of SD-RoF-based Speculative Cloud Gaming System", *2023 International Conference on Emerging Technologies for Communications*, pp. 1–1, November 2023.

Other Conference Papers

1. Takuya Fujiwara, Shunpei Yamaguchi, Takumasa Ishioka, Ritsuko Oshima, Jun Oshima, Kazuhiro Kizaki, Takuya Fujihashi, Shunsuke Saruwatari, Takashi Watanabe, "An IoT Sys-

tem for Collaboration Analytics in Hybrid Learning Environments", *IEEE International Conference on Advanced Learning Technologies (ICALT)*, pp.1–2, 2023.

2. Tsubasa Okamoto, Takumasa Ishioka, Ryota Shiina, Tatsuya Fukui, Hiroya Ono, Takuya Fujihashi, Shunsuke Saruwatari, Takashi Watanabe, "Edge-Assisted Multi-User 360-Degree Video Delivery", *IEEE Consumer Communications and Networking Conference 2023*, pp.1–6, 2023.

3. Shunsuke Akama, Takato Motoo, Takumasa Ishioka, Takuya Fujihashi, Shunsuke Saruwatari, Takashi Watanabe, "Deep Reinforcement Learning Model Design and Transmission for Network Delay Compensation in 3D Online Shooting Game", *IEEE Consumer Communications and Networking Conference 2023*, pp.1–6, 2023.

4. Kazuki Aiura, Takumasa Ishioka, Ryota Shiina, Tatsuya Fukui, Tomohiro Taniguchi, Kazuhiro Kizaki, Takuya Fujihashi, Takashi Watanabe, Shunsuke Saruwatari, "Design and Prototype Implementation of SD-RoF Networks", *The 13th International Conference on Mobile Computing and Ubiquitous Networking (IEEE ICMU'21)*, pp.1–6, 2021.

Abstract

In recent years, cloud gaming has gained significant attention in the gaming sector. Cloud gaming systems allow players to enjoy high-quality gaming experiences from anywhere without expensive hardware. However, a notable issue with cloud gaming systems is the degradation in perceived quality, especially regarding response latency. Response latency, the time it takes for a player's actions to reflect in the game visuals, is critical in fast-paced action games or those requiring quick reflexes. Increased latency can significantly reduce a player's immersion, potentially degrading the gaming experience. In modern games where real-time interaction is critical, effectively dealing with this delay is essential. This paper introduces speculative execution as an innovative strategy to reduce response latency in cloud gaming systems and examines its practical application. Speculative execution predicts potential player inputs, pre-computes the outcomes, and provides immediate responses to the actual player actions. However, speculative execution faces increased computational costs and traffic spikes. This paper proposes a comprehensive examination of speculative execution implementation, including the introduction of pattern reduction methods, traffic reduction strategies, and network architecture proposals. These approaches aim to address the challenges and enhance the feasibility of speculative execution in cloud gaming.

Chapter 3 introduces a novel pattern reduction method for speculative execution in cloud gaming. Pattern reduction is crucial in speculative execution for significantly reducing the number of frames to render, which simultaneously cuts computational costs and network traffic. The proposed method utilizes a temporal pattern analysis (TPA) method and an LSTM-based pattern prediction (LBPP) method with a bitfield representation of input signals to analyze player input logs, reducing the speculative processing of input patterns. The experiment involved ten male participants aged 20-30 years, using their input logs from playing popular games like Street Fighter V, Grand Theft Auto V, and Overwatch 2. The results demonstrated that TPA and LBPP significantly reduced the number of speculative execution patterns compared to the CloudHide-based (CHB) speculative approach, which generates and transmits game frames for all possible input patterns. Notably, LBPP maintained a nearly constant number of patterns even as the number of frames processed speculatively increased. This effect was consistent across various game titles, indicating the adaptability of the proposed methods. The experimental outcomes,

consistent across multiple game titles, confirmed that the proposed method effectively curbs the exponential increase in pattern numbers in speculative processing. Furthermore, traffic fluctuation analysis demonstrated that the proposed method effectively minimizes traffic while maintaining the necessary frame rate. These experimental results highlight the effectiveness of the proposed method in suppressing the exponential increase in speculative patterns, thereby enhancing the quality of the cloud gaming experience.

Chapter 4 proposes a traffic reduction method tailored to speculative execution in cloud gaming systems to mitigate network latency. The proposed method introduces a novel approach of dividing each frame into tiles and transmitting only those tiles with changes. This technique circumvents the limitations of traditional video compression methods in speculative approaches, achieving more efficient data transfer. We tested the traffic reduction effectiveness using game titles such as Valorant, Genshin Impact, and Street Fighter V. The results showed significant traffic reductions while maintaining high-quality video output with an SSIM (Structural Similarity Index Measure) of approximately 0.98. The reductions were about 35 % in Valorant, 24 % in Genshin Impact, and 53 % in SFV. The proposed method significantly reduced data traffic in speculative execution without relying on conventional video compression techniques.

Chapter 5 focuses on deploying cloud gaming systems on mobile networks, proposing a novel network architecture named Software Defined Radio over Fiber (SD-RoF). SD-RoF integrates analog RoF (Radio over Fiber) technology with RoF switching, enabling direct streaming of wireless signals from mobile devices to the network, significantly reducing physical latency. SD-RoF features two key functionalities: elastic wireless service and elastic bidirectional passthrough. The study presents a cost estimation and experimental evaluation of the prototype implementation of the elastic wireless service and a performance evaluation and demonstration of the elastic bidirectional passthrough. In the experimental evaluation of the elastic wireless service prototype implementation, we assessed the communication performance of each device. At the SD-RoF edge, we achieved a maximum signal reduction of approximately 54dB for unmodulated continuous waves and about 45dB for IEEE 802.15.4 data packets. The performance evaluation of the bidirectional passthrough assessed Packet Error Rate (PER), Round-Trip Time (RTT), and protocol adaptability. The reduction of loop signals at the SD-RoF edge improved communication quality. Additionally, we confirmed that direct communication is possible over approximately 20,000 meters using IEEE 802.15.4 and around 3,000 meters with Wi-Fi.

Through a holistic approach that includes exploring speculative execution and innovative network architecture, the study aims to improve the perceived quality of cloud gaming systems on mobile networks. The research offers critical insights for future innovations in mobile network technology. Our work can revolutionize gaming modalities and locales and impact other network-based interactive applications, such as remote driving technologies and telemedicine, including remote

surgery. The findings of our study highlight the broader implications and potential applications in enhancing real-time interactivity over networks.

Contents

1 Introduction **13**

2 Related Work **16**

2.1 Related Work on Network Techniques 16

2.1.1 Low-latency Network Techniques 16

2.1.2 Edge Computing 18

2.2 Optimization Techniques 20

2.2.1 Dynamic Video Quality Adjustment 20

2.2.2 Data Compression Techniques 21

2.2.3 Cloud Resource Allocation Optimization 23

2.2.4 Network Protocol Optimization 24

2.3 Related Work on Development Techniques 27

2.3.1 Cloud-native System Development 27

2.3.2 Latency Mitigation using Deep Reinforcement Learning 29

2.3.3 Speculative Execution on Cloud Gaming 31

3 Speculative Execution: Pattern Reduction Method **33**

3.1 Introduction 33

3.2 Proposed Method 35

3.2.1 Overview 35

3.2.2 Bit-Field-Based Pattern Reduction 36

3.2.3 Speculative Rendering 42

3.3 Evaluation 42

3.3.1 Experimental Measurements 42

3.3.2 Effect on the Number of Game Frames 42

3.3.3 Effect of Pattern Reduction Methods 43

3.3.4 Discussion about Trained Models 46

3.3.5 Experimental QoE Evaluation 47

3.4 Conclusion . . . 50

4 Speculative Execution: Dedicated Video Transmission Method 52

4.1 Introduction . . . 52
4.2 Proposed Method . . . 53
4.2.1 Overview . . . 53
4.2.2 Tile-wise Delta Detection . . . 54
4.2.3 Number of Speculative Frames Relative to Computational Capability . . . 56
4.3 Rate-distortion Optimized Speculative Frame Coding . . . 56
4.4 Evaluation . . . 58
4.4.1 Traffic Reduction in Commercial Games . . . 58
4.4.2 Feasibility . . . 62
4.5 Conclusion . . . 63

5 Low-latency Cloud Gaming System Architecture 65

5.1 Introduction . . . 65
5.2 SD-RoF: Software-Defined Radio over Fiber . . . 67
5.2.1 Architecture Overview . . . 67
5.2.2 RoF . . . 68
5.2.3 SD-RoF Switch . . . 69
5.2.4 SD-RoF Edge . . . 70
5.2.5 Elastic RoF Module . . . 70
5.2.6 Analog Cancellator . . . 71
5.3 Evaluation . . . 73
5.3.1 Performance Analysis . . . 73
5.3.2 Cost Estimation by Prototype Implementation . . . 77
5.3.3 Experimental Evaluation by Prototype Implementation . . . 78
5.3.4 Demonstration of Elastic Bidirectional Passthrough . . . 82
5.4 Conclusion . . . 90

6 Conclusion 91

List of Figures

3.1 Architecture of the Proposed Cloud Gaming System 35
3.2 Threading process in cloud gaming servers with speculative execution for two future frames. Here, the cloud gaming server has transmitted the user's game frames for $t+1$ at $T=t-1$ and t at $T=t-2$ to the user. 36
3.3 Overview of the TPA 37
3.4 Proposed bit-field-based TPA. In this case, the analysis is carried out for two future frames. 38
3.5 The overview of LBPP 39
3.6 The number of game frames rendered and transmitted in the existing and proposed methods. The possible input patterns in each frame are N, and speculative execution is used for two future frames. 41
3.7 Number of rendered game frames for speculative frames. The average framerate is set to over 57 fps for all methods. (a) compares CHB and the proposed method in OW2. (b) compares the proposed method across all games. 43
3.8 SFV: A comparative analysis between CHB and the proposed methods (a), alongside A comparative analysis between the proposed methods (b). 44
3.9 OW2: A comparative analysis between CHB and the proposed methods (a), alongside A comparative analysis between the proposed methods (b). 44
3.10 GTA: A comparative analysis between CHB and the proposed methods (a), alongside A comparative analysis between the proposed methods (b). 45
3.11 A comparison of the proposed methods with 1, 3, and 5 frames speculatively executed in OW2. 45
3.12 The average frame rate when the same LBPP-trained model is applied to each user. 46
3.13 Composition ratio of input experience quality in QoE MoS 49

4.1 Architecture of the Proposed Cloud Gaming System 54
4.2 Data Flow between Server and Client 54
4.3 Overview of the Proposed Tile-wise Delta Detection 55

4.4 Valorant snapshots comparing original and proposed methods. The proposed method's SSIM index and reduction ratio are shown, highlighting the efficiency and quality preservation. 58
4.5 Genshin Impact snapshots showing the effectiveness of the proposed method. The SSIM index and reduction ratio indicate significant improvements in performance and quality. 59
4.6 SFV snapshots comparing the original and proposed techniques, with a focus on the SSIM index and reduction ratio, demonstrate the effectiveness of the proposed method. 59
4.7 Traffics relative to SSIM index. To evaluate traffic based on the SSIM index, we measured the traffic and average SSIM index for each tiling by varying the Hamming distance threshold τ value from 0 to 64 for the original video frame. 60
4.8 The perceived network delay as a function of network delays with the different number of potential input patterns in each frame. 62
4.9 The perceived network delay as a function of the total clock speed, where the network delay is assumed to be 25 ms. 63

5.1 Comparison of existing and proposed systems 66
5.2 SD-RoF architecture 67
5.3 SD-RoF Switch 69
5.4 SD-RoF Edge 70
5.5 Elastic RoF Module 71
5.6 Frequency Converter 72
5.7 Radio signal loop between two SD-RoF edges 73
5.8 Analog cancellation of radio signal loop between two SD-RoF edges 74
5.9 Configuration of the analog cancellator 74
5.10 SD-RoF network evaluation model 75
5.11 Number of fiber optic cables relative to the number of interconnections 75
5.12 Experiment environment of elastic RoF module 79
5.13 Frequency conversion and multiplexing of OFDM signals using elastic RoF Module 80
5.14 Evaluation environment for the self-interference cancellation performance of the analog cancellator 81
5.15 Measured cancellation performance of analog cancellator 81
5.16 Evaluation environment with analog cancellator 82
5.17 Evaluation environment without analog cancellator 83
5.18 Comparison of PER with and without the analog cancellator 83

5.19 Comparison of the measured RTT and theoretical RTT derived from equation (5.8) 85
5.20 The effect of coverage class value 87
5.21 Frequency conversion and multiplexing of OFDM signals using elastic RoF Module 89
5.22 Waveform during the experiment 89

List of Tables

2.1 Major research topics on response time and traffic challenges in cloud gaming . . . 17

3.1 PC specifications . . . 42
3.2 The default constant coefficients of Eq. (3.13) . . . 47
3.3 Probabilities . . . 47

4.1 Relationship between SSIM and MOS [1] . . . 58
4.2 The SSIM index and traffic reduction ratio in each commercial game. The proposed method divided each video frame into 3×3 tiles and set τ=0 for tile-wise delta detection. . . . 58

5.1 Parts of elastic RoF module . . . 78
5.2 Parts of analog cancellator . . . 78
5.3 Basic characteristics of elastic RoF module . . . 79
5.4 The protocol applicability for SD-RoF . . . 86

List of Algorithms

1 The algorithm of the LBPP 40
2 Conversion of QoE expressed in R scale to MOS scale 47

Chapter 1

Introduction

Cloud gaming has emerged as a significant technological innovation in the gaming industry, advancing in parallel with the evolution of cloud computing. This field has made notable advancements since the introduction of GamingAnywhere, the first open-source cloud gaming system, in 2015 [2]. The development of mobile networks, particularly 4G and 5G, has effectively addressed longstanding network challenges. The COVID-19 pandemic further amplified the demand for cloud gaming as more users turned to digital entertainment options. The market for cloud gaming is experiencing rapid growth, projected to expand at an average annual rate of 48.20 % from 2023 to 2032 [3].

Cloud gaming's appeals are three key features: reducing the need for high-performance hardware, allowing for high-quality gaming on devices like smartphones; providing immediacy and convenience by eliminating downloads and installations; and offering cost-effective subscription models, granting access to a wide array of games across different genres from a single device. On the other hand, response delay remains a significant challenge in cloud gaming. The delay affecting a player's actions within a game is especially problematic in genres like fighting games, FPS, and rhythm games, which substantially impact the gaming experience compared to local play. Affecting both vision and control synchronization is a crucial area of research, as reducing response delay is vital for enhancing the gaming experience.

Traditional approaches to addressing response delay have primarily reduced propagation delay through edge computing and mitigating processing and transmission delays via adaptive bitrate control. Edge computing aims to shorten data transmission distances by processing data closer to the user, while adaptive bitrate control dynamically adjusts bitrates according to network conditions, ensuring efficient data transmission. Additionally, some games employ the object design to mitigate the impact of response delay. These include altering the size of objects in games to make the delay less perceptible to players and developing cloud-native games optimized for cloud-based play. Such strategies have shown effectiveness in improving the cloud gaming experience. However, these conventional methods face limitations, particularly in games requiring high-speed and real-time responses.

Some studies have proposed a new approach called speculative execution to address the traditional limitations. This technique involves predicting potential player actions and pre-computing various game scenarios based on these predictions, transmitting the appropriate visuals in advance. Theoretically, speculative execution could achieve zero latency from the user's perspective, offering a fundamental solution to the response delay problem in cloud gaming. However, full implementation poses challenges, such as immense computational costs and increased traffic volume due to the need to transmit visuals for all possible action patterns.

Our research introduces key innovations that address the inherent limitations of conventional speculative execution methods. We propose a comprehensive approach that enhances the predictive accuracy of player actions and significantly mitigates the computational and traffic-related challenges associated with speculative execution. By refining the process of pre-computing game scenarios and optimizing the transmission of frames, our methodology aims to achieve an efficient balance between performance and resource usage. These approaches are crucial for realizing the full potential of cloud gaming, especially in addressing the critical issues of latency and traffic volume that have historically constrained conventional speculative execution techniques. The subsequent chapters delve into how our approach uniquely solves these problems.

Chapter 3 introduces a pattern reduction method designed to mitigate these challenges. This method utilizes preliminary analyses based on users' operation logs, using pattern extraction with time series analysis or LSTM network-based predictions to identify the high-probability input patterns. This approach significantly enhances the efficiency of speculative execution. We aim to reduce the number of predicted patterns to a manageable few while maintaining high estimation accuracy, thereby controlling computational costs and curbing the exponential growth of multi-frame predictions.

Chapter 4 proposes a novel approach to address the increased traffic associated with speculative execution. Traditional video compression methods are unsuitable for speculative cloud gaming systems. We have developed a technique that involves splitting each frame into tiles and transmitting only those tiles that differ due to user operations. This method efficiently reduces traffic without relying on conventional video compression technologies, providing an effective solution to the problem of traffic increase proportional to the number of patterns in speculative cloud gaming systems.

Chapter 5 delves into the challenges and solutions of deploying cloud gaming systems in mobile networks. As mobile devices like smartphones constitute the primary market for cloud gaming systems, the issue of response delay is more pronounced in these networks than in wired networks. We propose a novel network architecture utilizing Radio over Fiber (RoF) technology. This technology enables the direct streaming of mobile device wireless signals into the network, significantly reducing the processing delays required for network routing. In the proposed network architecture,

speculative execution in cloud gaming systems is only subject to propagation delay, which can radically reduce latency while managing the increased traffic volume with RoF's high bandwidth capabilities.

Our research not only tackles the technical challenges but also paves the way for new possibilities in cloud gaming, bringing it closer to the immediacy and responsiveness of local gaming. By minimizing network delay and optimizing system performance, we enable users to enjoy high-quality gaming experiences, regardless of their location or the capabilities of their devices. The proposed speculative cloud gaming system is a pioneering approach, adaptable to various game genres and devices, and represents a significant advancement in the cloud gaming landscape. Our research contributes to a holistic solution that enhances the cloud gaming experience, from advanced pattern recognition algorithms to innovative network architectures.

Chapter 2

Related Work

This section comprehensively examines the critical research topics addressing response time and cloud gaming traffic challenges. Table 2.1 shows major research topics on cloud gaming response time and traffic challenges. These topics span a range of techniques, from network advancements to innovative game development strategies, each playing a crucial role in enhancing the cloud gaming experience. We will describe how advancements in network technologies, optimization methods, and system and gameplay enhancements collectively contribute to mitigating latency, improving performance, and ensuring a seamless gaming experience for users.

2.1 Related Work on Network Techniques

Network techniques in cloud gaming are fundamental to addressing the inherent challenges posed by high latency and variable bandwidth, which can significantly affect the user experience. This section surveys the advancements in network technologies and their application in the context of cloud gaming. [4]

2.1.1 Low-latency Network Techniques

Choy et al. [5] determined that a total latency of around 100 ms is a critical threshold for cloud gaming, with 20 ms of this latency attributed to playout and processing delays. The acceptable network latency should not exceed 80 ms to avoid negatively impacting the user experience. The advent of 5G and impending 6G networks promises transformative changes for cloud gaming, offering substantially lower latencies and higher throughput [6]. Research has shown that these technologies can achieve latencies below 20 milliseconds [7], which is crucial as it nears the threshold for imperceptibility to human reflexes in interactive applications [8]. The lower latency afforded by these networks is essential in delivering a seamless gaming experience akin to playing on a local console or PC. Additionally, the high throughput ensures the streaming of high-quality game visuals without compromising fidelity or causing interruptions.

Table 2.1: Major research topics on response time and traffic challenges in cloud gaming

Techniques	**Research Topics**	**Description**
Network Techniques	Low-latency Network Techniques	Development and application of advanced network technologies such as 5G/6G and software/hardware optimizations for achieving sub-20 ms latencies.
	Edge Computing	Utilization of edge computing to reduce data travel distance, process games closer to players, and manage QoS, thereby mitigating latency and traffic concerns.
Optimization Techniques	Dynamic Video Quality Adjustment	Adaptive adjustment of video quality in response to bandwidth fluctuations to ensure smooth gameplay.
	Data Compression Techniques	Implementation of advanced image, audio, and game asset compression methods for server-side optimization to reduce data transfer volume and enhance performance.
	Cloud Resource Allocation Optimization	Dynamic allocation and deallocating cloud resources in real-time based on demand to optimize performance and cost.
	Network Protocol Optimization	Development of gaming-specific network protocols for optimized data packet delivery and reduced latency.
System and Gameplay Enhancements	Cloud-native System Development	Designing games specifically for cloud environments with a focus on scalability, cloud-based features, and server-side asset optimization.
	Latency Mitigation using Deep Reinforcement Learning	Deep reinforcement learning to optimize various aspects of cloud gaming, including resource allocation and predictive algorithm for speculative execution.
	Speculative Execution on Cloud Gaming Servers	Implementing predictive algorithms, enhanced with deep reinforcement learning, for pre-rendering game frames based on potential user actions to minimize response time.

Beyond the physical network layer, substantial efforts have been made in software and hardware optimization to reduce latency. For example, Amiri et al. [9] developed Network functions virtualization (NFV) to optimize the flexibility and responsiveness of network services. Gouareb et al. [10] discuss optimizing NFVs placement and routing across physical hosts to minimize overall latency, a critical factor in cloud gaming networks.

Specialized protocols also play a significant role in streamlining data transfer and reducing overhead. For instance, using User Datagram Protocol (UDP) over Transmission Control Protocol (TCP) for real-time game data transmission can reduce latency. UDP's less stringent approach to data transmission is better suited for the time-sensitive nature of gaming, where receiving newer packets is often more crucial than ensuring every packet is received [2]. Parsec [11], a cloud gaming service, exemplifies this by developing its networking protocol, BUD, based on UDP. This protocol is tailored for low latency and interactive game streaming, offering the essential reliability of TCP but with customized congestion control to minimize latency.

Additionally, hardware-level optimizations, such as advanced network interface cards and optimized routing equipment, are pivotal in minimizing latency in cloud gaming [12]. Network optimization also encompasses deploying game session management tools that dynamically adjust

the server load based on real-time network performance data. Amiri et al. [13] proposed an innovative method for optimizing bandwidth allocation in home networks to enhance the Quality of Experience (QoE) in online gaming, primarily cloud gaming. The Game-Aware Resource Manager (GARM) intelligently assigns network resources based on each application's requirements, aiming to reduce latency and improve video quality for gaming without negatively impacting other applications. Experimental results demonstrate that GARM significantly reduces network delay and enhances video quality, improving the gaming experience. Furthermore, Lien et al. [14] explored ultra-low-latency methodologies in Heterogeneous Cloud Radio Access Networks (H-CRAN), which has significant implications for cloud gaming, where latency is a crucial aspect of the user experience. Techniques like open-loop radio access and cognitive radio resource management proposed in the paper can effectively reduce network latency and signaling overheads, essential for real-time responsiveness in cloud gaming. Additionally, the paper's focus on social data cache-based routing/paging schemes can optimize network routing and packet forwarding, further enhancing cloud gaming performance by ensuring faster and more efficient data transmission. While not exclusively centered on cloud gaming, the paper's insights and solutions are highly relevant and adaptable for improving network performance in cloud-based gaming systems.

These tools are part of a broader suite of cloud gaming management systems that monitor network conditions, player count, and server performance, adjusting resources in real-time to maintain optimal gaming conditions. This dynamic resource allocation is vital in handling peak load times and ensuring each game session receives the necessary computational resources without delay. Basiri and Rasoolzadegan [15] proposed an optimization framework for cloud gaming systems to minimize operational costs while ensuring ultra-low latency to satisfy QoE requirements. It introduces a comprehensive delay model that accounts for various sources of delay in cloud gaming and formulates an optimization problem to efficiently allocate virtualized resources in cloud data centers. They proposed a cloud gaming platform balancing cost efficiency with the imperative of providing crucial low-latency services.

In summary, developing and implementing low-latency network techniques in cloud gaming is multi-dimensional, involving advancements in network technology, protocol optimization, hardware enhancements, and intelligent resource management. Together, these efforts are critical in realizing the full potential of cloud gaming as a high-performance, accessible, and responsive platform.

2.1.2 Edge Computing

Research in Edge Computing, particularly in cloud gaming, has been crucial in addressing latency challenges, a critical factor in gaming experiences. The decentralization of the computing infrastructure and its proximity to the end-user are crucial elements in this approach. For instance, studies have applied mobile edge computing technologies to reduce response times, with efforts to

decrease the distance between the cloud gaming server and the user terminal to reduce propagation delay. Zhang et al. [16] introduces a framework that leverages Mobile Edge Computing to enhance cloud gaming by offloading computation-intensive tasks like game rendering to edge nodes. This approach significantly reduces network delay and bandwidth usage, improving the QoE for gamers. The paper also discusses future enhancements, including a crowdsourced edge network and a blockchain-based incentive mechanism, to optimize cloud gaming infrastructure further. Amiri et al. [17] introduce a bi-objective optimization method to minimize delay and maximize bandwidth utilization in cloud gaming data centers, employing an analytic hierarchy process-based game-aware routing scheme. This approach considers the specific requirements of different game types, reducing end-to-end delay and balancing game flows within the network, which is crucial for high QoE in cloud gaming.

A prime example of effective edge computing implementation in cloud gaming is NVIDIA's GeForce Now [18]. This service strategically places its servers worldwide, aiming to minimize the distance data travels by being as close to end-users as possible. This geographical distribution of servers is essential in reducing latency and enhancing the gaming experience [19]. Furthermore, NVIDIA's data centers, equipped with high-performance hardware optimized for game streaming, ensure high-quality gaming experiences with minimal lag, even for users without gaming-grade hardware [20]. NVIDIA, which develops cloud gaming systems, is adapting to expanding server locations. By continuously evaluating and expanding server locations based on user demand and network performance data, NVIDIA ensures that player populations and network conditions are adequately catered to across different regions and times, providing a consistently high-quality gaming experience [21].

In addition to server placement, improving the Quality of Service (QoS) for cloud gaming has been a focal point in network techniques, particularly in edge computing [22]. Prioritizing gaming traffic over other data types is crucial for maintaining consistent gameplay performance. These measures are vital in maintaining a high service quality in cloud gaming. Chen et al. [22] proposed insights into implementing and optimizing QoS in cloud gaming environments, including cognitive resource optimization. Madiha et al. [23] proposed the evaluation of QoS and QoE within Fog Computing, and Lin and Shen [24] proposed systems that reduce response latency and bandwidth consumption in cloud gaming.

In summary, edge computing in cloud gaming is not just about server placement; it also involves optimizing network traffic and resource allocation to enhance the gaming experience. The case of GeForce Now exemplifies how strategic server placement, combined with QoS enhancements, can effectively reduce latency and improve overall game responsiveness in a cloud gaming environment.

2.2 Optimization Techniques

The optimization techniques employed in cloud gaming improve the efficiency and performance of game delivery and execution. Essential methods include video and data compression, cloud resource allocation, and network protocol design. Each area is crucial in minimizing latency, reducing bandwidth consumption, and enhancing the overall QoS.

2.2.1 Dynamic Video Quality Adjustment

The dynamic adjustment of video quality, commonly known as adaptive bitrate streaming, is central to maintaining gameplay fluidity under varying network conditions [25–27]. This technique dynamically adjusts the video encoding rate based on bandwidth availability, ensuring that games remain playable even when network performance fluctuates. Adaptive bitrate streaming operates by continuously monitoring network conditions and adjusting the quality of the video stream accordingly, a focus of platform optimization research including dynamic video quality adjustment [28]. When bandwidth is high, the streaming quality increases, providing a richer and more detailed gaming experience. Conversely, when bandwidth is limited, the system needs to reduce the video quality to prevent latency spikes and buffering, thus maintaining a consistent gaming experience [29].

Research in this domain has produced various sophisticated algorithms capable of predicting bandwidth availability. These algorithms use historical data, real-time network metrics, and sometimes machine learning techniques to forecast network conditions accurately. By preemptively adjusting video quality, these systems can mitigate potential latency issues before they affect the gamer's experience [30]. Gharsallaoui et al. [31] proposed the importance of adaptive streaming in enhancing the performance of cloud-based mobile video games. It presents a novel streaming method that dynamically adjusts to various network conditions, ensuring efficient data transmission and improved gaming experience. The study highlights the significance of this approach in maintaining high-quality gameplay on mobile devices, even under fluctuating network environments typical of mobile gaming scenarios.

One of the critical challenges in dynamic video quality adjustment is balancing the trade-off between video quality and gameplay smoothness. High-quality video enhances the gaming experience but requires more bandwidth, while lower quality ensures smoother gameplay at the expense of visual fidelity. One of the critical challenges is balancing video quality and gameplay smoothness. Mori et al. [32] introduced an HTTP adaptive streaming method that enhances the QoE by considering the motion intensity of video scenes. This method allows a streaming client to adaptively select the quality of the following video segment based on the motion intensity, choosing higher quality for high-motion scenes and lower quality for low-motion scenes to optimize streaming

performance and prevent stalling. The effectiveness of this approach is demonstrated through implementation and testing on an actual system, showing an improvement in QoE compared to conventional streaming methods.

In summary, dynamic video quality adjustment through adaptive bitrate streaming is vital in delivering a stable and enjoyable cloud gaming experience. It represents a sophisticated intersection of network technology, video encoding, and predictive analytics, all working together to ensure that games are playable and visually pleasing under varying network conditions.

2.2.2 Data Compression Techniques

Efficient data compression is vital in cloud gaming for reducing the volume of data transferred between cloud servers and end-users, thereby reducing response time and bandwidth requirements. Optimizing data transmission without compromising game quality is a significant challenge in a cloud gaming system where real-time interaction is crucial.

Traditional compression techniques in cloud gaming involve adapting various image and audio compression methods to suit the requirements of interactive gaming over the network. The differential coding, such as that used in H.264/AVC [33] and H.265/HEVC [34], achieves traffic reduction by encoding and transmitting only the changes between the current and previously encoded frames. The advancements in codec technologies benefit cloud gaming video quality, which can remain sufficiently high even at lower bitrates [28]. Storch et al. [35] presented a comprehensive evaluation of parallel video coding in HEVC using tile-based partitioning, focusing on the impact of uniform and non-uniform tiling patterns on encoding speedup and coding efficiency. The study demonstrates that while uniform tiling's effectiveness varies with video sequences and encoding profiles, transitioning to optimal non-uniform tiling can significantly enhance speedup without compromising coding efficiency. The research underscores the potential of tailored tiling strategies in efficiently exploiting parallel processing in HEVC encoding. Additionally, Chuah et al. [36] proposed advanced lossless compression algorithms to provide efficient compression without quality loss, making them more suitable for cloud gaming applications. The type of game, the importance of visual fidelity, and the network conditions affect the better choice between lossy and lossless compression. Some cloud gaming systems may employ adaptive compression methods, which switch between lossy and lossless techniques based on real-time assessment of network conditions and the specific requirements of the game session [37].

Beyond cloud gaming, studies on traffic reduction in immersive video applications have utilized adaptive tile-based virtual reality (VR) video transmission methods. Tile-based video transmission schemes divide the entire video frame into multiple tiles, which allows for video coding parallelization to decrease coding delay and provides fine-grained quality control by transmitting only the viewport area actively within the user's view [38–41]. Another study aimed at reducing traffic for

multi-view video coding has leveraged disparity prediction and compensation to remove inter-view redundancy, thus optimizing bandwidth use across multiple viewpoints [42].

Besides, advanced compression methods increasingly leverage machine learning to enhance compression rates without sacrificing the quality of game visuals and audio. Machine learning algorithms analyze gaming data to identify patterns and optimize compression in real-time. Some studies proposed traffic reduction methods based on the user's perspective information to prevent the user's perceptual quality degradation. Hegazy et al. [43] improved the perceptual quality by maximizing the video quality of regions of interest within each frame, focusing on what players are most likely to notice. Sabat et al. [44] determined the video quality based on gazing points at objects within the game and rendered the most visually critical areas with higher quality. Illahi [45] proposes a foveated video encoding method to determine the quality within each game video frame based on the user's gaze information, taking advantage of the human visual system's property of lower resolution toward the periphery of the field of view. These methods can determine which parts of a frame are most likely to draw the player's attention and compress less important areas more aggressively. These machine learning-based compression techniques adapt dynamically to the streamed content, allowing for more efficient bandwidth use. They are instrumental in highly variable network conditions, as they can adjust compression rates on the fly to maintain a balance between quality and smooth gameplay.

Chi et al. [46] evaluated and compared three parallelization strategies for High-Efficiency Video Coding (HEVC): tiles encoding, wavefront parallel processing, and a novel approach named Overlapped Wavefront (OWF). While each method has distinct advantages, OWF outperforms others in speedup and efficiency, especially at higher video resolutions. The study critically analyzes these strategies, highlighting their implications for enhancing video coding in increasing demand for higher-resolution content. Chen and El-Zarki [47] propose a novel cloud gaming framework that dynamically adjusts game graphics quality to optimize network bandwidth and reduce latency. It utilizes collaborative rendering and progressive meshes to balance computational tasks between the cloud and the client device, catering to the client's hardware capabilities and network conditions. This approach, backed by extensive performance evaluations, demonstrates its effectiveness in enhancing cloud gaming experiences, particularly regarding bandwidth efficiency and reduced interaction latency.

In summary, data compression techniques in cloud gaming are an amalgam of traditional methods and cutting-edge innovations. The practical application of these techniques is crucial in delivering a seamless gaming experience, particularly in scenarios where bandwidth is a limiting factor. As cloud gaming continues to evolve, so will the methods for efficiently compressing game data, ensuring that high-quality gaming experiences are accessible even over constrained network conditions. We proposed a tile-wise delta detection method for simultaneously sending multiple game

video frames. The tile-wise delta detection method uses a hash function-based similarity prediction to detect changes between frames, reducing computational cost for encoding.

2.2.3 Cloud Resource Allocation Optimization

Optimizing the allocation of cloud resources is a critical area in cloud gaming research. Effective management of resources such as CPU, GPU, and memory is essential for ensuring that cloud gaming platforms can deliver high performance while being cost-effective. Han et al. [48] presented the importance of optimizing virtual machine placement to meet diverse QoE demands of players, proposing a scalable, game theory-based distributed algorithm that efficiently manages cloud resources in response to real-time gaming demands, thus ensuring a high-quality gaming experience while maintaining cost-effectiveness.

The core of cloud resource allocation lies in dynamically allocating and deallocating resources based on real-time demand. Sophisticated algorithms are developed for this purpose, enabling the cloud gaming platform to quickly respond to changing player numbers and game complexity. For instance, during peak gaming hours or for graphically intensive games, these systems can allocate additional resources to maintain smooth gameplay. Conversely, resources can be scaled back during off-peak hours or for less demanding games to conserve energy and reduce costs. Cai et al. [49] discussed how a cognitive, component-based system in cloud gaming could significantly enhance the quality of experience through dynamic resource allocation, effectively addressing challenges posed by diverse end-user devices and fluctuating network qualities.

Predicting player demand is a challenging yet crucial aspect of resource allocation. Research in this field has introduced various models that simulate gaming workloads to improve the accuracy of these predictive systems. These models consider historical data, game popularity, player behavior patterns, and events like game launches or updates to forecast resource needs. By accurately predicting demand, cloud gaming services can pre-emptively adjust their resource allocation, ensuring that players consistently receive a high-quality gaming experience. Deng et al. [50] proposed the importance of managing server allocation in multiplayer cloud gaming, presenting several heuristic algorithms to optimize server and bandwidth resource allocation under varying costs and real-time interaction requirements.

Another important aspect of resource allocation is load balancing, which involves distributing gaming workloads across multiple servers to optimize performance and reduce latency. Effective load balancing avoids reduced performance and increased latency. Generally, techniques such as geographic load balancing determine the route of a player's connection to the closest server with available capacity. Additionally, Chen et al. [47] discussed balancing trade-offs between network bandwidth, visual quality, and interaction latency in cloud gaming. It introduces a framework employing collaborative rendering, progressive meshes, and 3D image warping to address these

challenges, adapting to the varying capabilities of client hardware and network conditions.

Alongside performance optimization, cost management is a significant concern in cloud resource allocation. The cost of operating cloud gaming services is closely tied to resource usage, making efficient allocation key to profitability. Li et al. [51] argued for a balance between server running costs and software storage costs, proposing heuristic algorithms for server provisioning that consider game software distribution among servers, server types, and user demand patterns. This approach aims to minimize total operational costs while maintaining satisfactory gaming service for a large user base.

Luo and Claypool [52] propose an open-source, cloud-based gaming system, Uniquitous, designed to overcome the challenges in commercial cloud gaming. It emphasizes the system's ability to offer enhanced control over game content and cloud infrastructure and presents its performance evaluation in various configurations. The study underscores Uniquitous as a flexible platform for research and development in cloud gaming, highlighting its potential to optimize game performance and address network and server-related issues in cloud-based gaming environments. Cai et al. [53] argue for a novel cloud gaming system that leverages cognitive computing to adaptively balance game components between the cloud and user devices. It optimizes the gaming experience by dynamically adjusting to varying device capabilities, network conditions, and player behaviors. The paper emphasizes the system's ability to maintain high-quality gaming experiences across diverse environments, showcasing its effectiveness through empirical performance evaluations. Furthermore, Yami et al. [54] highlighted the effectiveness of incorporating game state as a criterion for network resource allocation in enhancing QoE. It introduces the state-aware resource allocation in Software-Defined Networks (SDN) architecture. It dynamically allocates network paths based on game state and network conditions, optimizing the gaming experience by better addressing game-specific requirements like latency and bandwidth.

In summary, cloud resource allocation in gaming involves a complex interplay between dynamic resource management, predictive modeling, load balancing, and cost optimization. As cloud gaming grows, developing more sophisticated and efficient resource allocation techniques will remain a central focus, aiming to provide an optimal balance between high performance and cost-effectiveness.

2.2.4 Network Protocol Optimization

Optimizing network protocols is crucial in reducing latency and enhancing the robustness of connections in cloud gaming. The ultimate goal of network protocol optimization in cloud gaming is to achieve a reliable and low-latency network. For example, a comprehensive approach combines the development of new protocols, the enhancement of existing ones, and the implementation of effective QoS strategies. These efforts significantly reduce the latency and jitter that can detract from the cloud gaming experience, ensuring that players have smooth and immersive gaming sessions.

The unique demands of interactive cloud gaming, such as the need for real-time responsiveness and high data throughput, have proposed gaming-specific transport protocols. These protocols optimize the order and timing of data packet delivery, prioritizing game-critical data to ensure smooth gameplay. For example, Amiri et al. [55] proposed that optimizing the allocation of gaming sessions to geographically distributed data centers using a hierarchical SDN architecture can significantly improve the QoE in cloud gaming. The proposed method effectively minimizes network delay by formulating and applying an Online Convex Optimization approach. It maximizes bandwidth utilization, enhancing the service provider's resource efficiency and the gamers' experience.

In addition to developing new protocols, optimizing existing network protocols is also a key strategy. For example, standard protocols like TCP and UDP better suit the needs of cloud gaming [56, 57]. The modifications include adjusting timeout intervals, optimizing congestion control mechanisms, or implementing more efficient error handling to ensure data integrity without significantly impacting latency.

Alós et al. [58] proposed that effective congestion control tailored for UDP can significantly enhance cloud gaming experiences, particularly by focusing on round-trip video latency. It proposes an approach that dynamically adjusts congestion control mechanisms based on latency observations, ensuring smoother gameplay and more efficient network utilization in cloud gaming scenarios. Wu et al. [59] presented an innovative method for streaming mobile cloud gaming content over TCP, utilizing adaptive forward error correction coding. This approach optimizes TCP for cloud gaming by improving reliability and reducing latency, enhancing the mobile device gaming experience. Faisal and Zulkernine [60] proposed a secure architecture for cloud communications using TCP/UDP, emphasizing the importance of security in cloud gaming networks. The paper argues that this architecture effectively mitigates security risks while maintaining the performance efficiency necessary for a high-quality cloud gaming experience.

Besides, implementing QoS prioritization techniques is essential to network protocol optimization in cloud gaming. These techniques ensure that gaming traffic is prioritized over other types of network traffic, leading to a consistent and high-quality gaming experience. Methods like network slicing allow the creation of dedicated network paths for gaming traffic, preventing congestion caused by other network uses. Liao et al. [61] presented a detailed survey of various network QoS methodologies. It argues for the critical importance of QoS in network communication, particularly in scenarios where network performance and reliability are paramount. Li et al. [62] presented an approach towards minimizing resource usage while ensuring QoS in cloud gaming. Using efficient resource allocation strategies can enhance the gaming experience without excessive resource consumption, thus balancing performance and resource management in cloud gaming environments. Balusamy et al. proposed an active implementation framework to enhance QoS in multiplayer cloud gaming systems. They argue that this framework significantly improves the gaming experience in

a shared network environment by ensuring stable and high-quality service, which is essential for multiplayer gaming scenarios.

Nammas and Quwaider [63] focused on evaluating the Quality of Service of cloud gaming systems efficiently. Through proper assessment and optimization of QoS, cloud gaming platforms can significantly enhance their performance, ultimately improving the overall gaming experience for users. Lindström et al. [64] analyze the impact of QoS factors on the QoE and in-game performance in cloud gaming. They demonstrate that 'frame age' is a crucial QoS metric influencing players' experience and performance, particularly under network conditions like latency and packet loss. The study underscores the importance of optimizing frame age and other QoS factors to enhance the gaming experience in cloud-based environments. Laghari et al. [65] presents a framework designed to improve the QoE in cloud computing services, particularly video streaming. The QoC framework integrates subjective user feedback and objective QoS data, enabling accurate QoE predictions and effective service management. This approach addresses key cloud computing challenges such as service degradation and SLA violations, aiming to align service delivery with user expectations and enhance overall user satisfaction.

Traffic shaping and priority queuing are also vital in managing cloud gaming traffic. By controlling the rate of traffic sent to the network and prioritizing packets based on their importance to game performance, these techniques help maintain a stable and responsive gaming environment. It is crucial for sharing network resources between various services and applications. Graff et al. [66] proposed that accurately identifying cloud gaming traffic at the network edge is crucial for implementing effective traffic management strategies in cloud gaming environments. It emphasizes the importance of efficient traffic shaping and priority queuing to ensure a high-quality gaming experience, particularly in edge computing scenarios where network resources are limited and highly valuable. Baldovino et al. [67] presents a comprehensive analysis of networking issues in cloud gaming, including the critical role of traffic shaping and priority queuing. It argues that these techniques are essential for managing cloud gaming traffic, reducing latency, and ensuring that game data packets are delivered efficiently and reliably, critical factors in maintaining an optimal gaming experience.

In summary, optimization techniques for cloud gaming are diverse and address both the content delivery aspects and the network performance challenges. The related work in this field is extensive and continually evolving, aiming to provide an efficient, cost-effective, and high-quality gaming experience on par with traditional local gaming setups.

2.3 Related Work on Development Techniques

Development techniques in cloud gaming include a range of strategies and methodologies to create games optimized for cloud environments. This section discusses various approaches to speculative execution as a critical area of this research.

2.3.1 Cloud-native System Development

Cloud-native system development focuses on designing games specifically for cloud environments. This development approach considers network variability, server-client architecture, and scalability from the game development stage. Lee et al. [68] demonstrate that not all games are equally suitable for cloud gaming due to varying degrees of real-time strictness in each genre, which impacts the QoE. Some studies have explored system adjustments to reduce the effect of extended network delay on the quality of experience, which is an essential consideration in cloud-native system development.

Yates et al. [69] introduced a model for optimizing cloud gaming systems by focusing on the timeliness of video frame delivery using an Age of Information (AoI) metric. The AoI is particularly relevant for low-latency mobile gaming and involves a detailed analysis of frame delivery and synchronization between the game server and mobile clients. The study emphasizes optimizing frame rate and lag synchronization to enhance the QoE in cloud-assisted gaming applications. Sabat et al. [70] focused on the challenges and solutions related to cloud gaming, particularly emphasizing the importance of efficient network management and resource allocation. It argues that in cloud gaming, ensuring low latency and high reliability of data transmission is crucial for providing a satisfactory gaming experience. Salay and Claypool [71] focused on the identification and management of cloud gaming traffic within edge computing environments. It argues that precise identification and efficient handling of cloud gaming data at the network edge is essential for enhancing the gaming experience, ensuring reduced latency, and optimizing network resource utilization.

Kim et al. [72] presented post-render warp techniques, adapted from virtual reality, to mitigate latency in remote-rendering systems, particularly for aiming tasks in cloud gaming. Their experiment with experienced gamers demonstrates that late-warp methods can significantly reduce the performance penalty caused by latency, eliminating up to 80 % of the aiming task performance degradation caused by 80 ms of added latency. Implementing these techniques in game streaming services could improve the competitive gameplay experience, especially in first-person shooters. Sabet et al. [73] proposed that while the type of game significantly affects user perception of delay and QoE in cloud gaming, the strategies players employ within these games do not. They demonstrate that user input characteristics vary between games but remain consistent across different

strategies within the game, suggesting that the chosen strategy does not significantly influence the perception of delay or overall gaming experience.

Specialized frameworks for cloud gaming development are integral to facilitating this process. These frameworks provide developers with tools and libraries optimized for cloud gaming, such as network-aware asset management and scalable multiplayer infrastructure. These frameworks often include APIs that allow easy integration of cloud-based features and facilitate the development of inherently scalable and network-efficient games. Research in this domain focuses on enhancing these frameworks' scalability, performance, and developer-friendliness. This approach also allows games to operate during performance degradation and cloud service failure, significantly enhancing game engine graphics development across heterogeneous platforms.

Buluman and Garraghan [74] proposed a framework for dynamic graphical rendering in cloud gaming, aiming to distribute game engines across cloud and client systems. It argues that this approach improves frame rates and fault tolerance and allows game developers to leverage diverse graphical APIs, enhancing the graphics development process for cloud gaming across various platforms. Iqbal et al. [75] introduced Dissecting Cloud gAming perFormance (DECAF), a methodology for systematically analyzing cloud gaming performance. In DECAF, cloud processing delays form the most critical factor in game delays. Cloud gaming platforms often struggle with bitrate delivery and maintaining high-quality video streams, especially under packet loss conditions. The study highlights the necessity of comprehensive measurement methodologies to understand and improve the performance of cloud gaming platforms. Giovanni et al. [76] propose the Cloud-Oriented Distributed Engine for Gaming as a solution to the limitations of traditional cloud gaming infrastructures, which often rely on monolithic game engines. They utilize a distributed architecture that leverages cloud and edge computing resources, aiming to enhance the gaming experience and reduce operational costs. It does this by partitioning game engines into interconnected modules placed strategically across the network, optimizing resource utilization and response latency to better balance player experience and cost efficiency.

Cloud gaming systems render games on the server side before streaming. Therefore, optimizing game asset storage and processing on the server is preferred. While these techniques do not directly affect the video stream sent to the client, they reduce the time to render game frames by optimizing how game assets are stored and processed on the server. It is crucial to minimize overall latency and improve the game's responsiveness. Amiri et al. [9] proposed that minimizing end-to-end latency is essential for enhancing the cloud gaming experience. It proposes a novel method involving a Lagrangian Relaxation time-efficient heuristic algorithm to optimize server and path selection within cloud gaming data centers. This approach aims to reduce overall delay, improve fairness among players, and cater to the varying delay sensitivities across different game genres, thereby enhancing the quality of experience for cloud gaming users.

2.3.2 Latency Mitigation using Deep Reinforcement Learning

Deep Reinforcement Learning (DRL) is a potent tool for mitigating latency in cloud gaming. The applications span several critical areas of cloud gaming service and service optimization. A key research focus is the development of DRL algorithms that are computationally efficient and can adapt in real-time to the changing conditions of cloud gaming environments. These models must balance accuracy and computational speed to enhance the gaming experience without introducing additional latency. In cloud gaming, DRL is pivotal for optimizing resource allocation across gaming sessions. By considering factors such as the number of active players, game demands, and infrastructure load, DRL models can dynamically distribute resources like CPU, GPU, and memory, maintaining high performance and reducing latency.

DRL also plays a crucial role in network traffic management. DRL algorithms can forecast high-traffic periods and adjust network configuration. Optimizing routing paths and managing data packet prioritization ensure seamless gameplay during peak times. Deng et al. [77] proposed an adaptive resource allocation strategy using DRL to address the challenges of server overload and player mobility in edge computing, demonstrating its effectiveness in improving system performance, including latency fairness and load balance. By situating computing resources closer to gamers, edge computing significantly reduces the distance data must travel, thus lowering latency.

Moreover, DRL's capability extends to continuously improving the gaming experience. Justesen et al. [78] reviewed the application of deep learning techniques in video game playing, focusing on various game genres and their unique challenges. It highlighted the advancements in deep learning and reinforcement learning algorithms, such as Deep Q-Networks and actor-critic methods, and their effectiveness in addressing challenges like large decision spaces and sparse rewards in games. The paper emphasizes the significant progress in applying deep learning to games and the ongoing challenges, particularly in environments with sparse feedback. Cloud gaming systems have the potential to gradually enhance the gaming experience by collecting and learning from a large amount of data from users. Romero-Mende et al. [79] argue for enhancing player engagement in video games using a deep learning-based system for dynamic difficulty adjustment (DDA). This system classifies players' skill levels in real-time and adjusts the game's difficulty accordingly to provide a balanced and personalized gaming experience. The study demonstrates the effectiveness of this approach, showing that the DDA system significantly improves game enjoyment and immersion, marking an essential advancement in player-centered game design.

Complementing these DRL applications is the field of input prediction, which serves as an active research area for latency concealment. Techniques like GGPO speculatively apply local inputs, aligning them with future actual inputs, a method proven effective in fast-paced fighting games where player prediction is possible [80]. Extensions of this approach, utilizing hidden

Markov models and AI agents, have been developed to improve input predictions in fighting games. Ehlert [81] investigates the potential for enhancing input prediction in online fighting games, currently limited by using a simple repeat-last-frame method in rollback netcode. The study proposes a new prediction model based on recurrent neural networks, trained and evaluated using a dataset of recorded player input sequences. While the new model slightly improves prediction accuracy over the existing method, it also incurs significantly higher computational and memory costs, suggesting limited applicability in enhancing current rollback netcode implementations. Zamith et al. [82] argue for using Hidden Markov Models (HMM) to balance game difficulty based on player behavior dynamically. It presents an innovative approach where the HMM algorithm adapts the game in real-time, learning from player patterns and actions. This method, validated through a Space Invaders clone experiment, demonstrates improved player engagement by adjusting game challenges to suit individual player experiences. Zuin and Macedo [83] proposed using HMMs to detect infinite combos—uninterruptible sequences of attacks. They showed that HMMs help identify straightforward combos. Simulatancly, they suggest comprehensively addressing combos in game design requires more specialized and adaptable modeling approaches.

However, these methods often focus narrowly on specific game genres. The current research trend is the application of DRL to AI development rather than input prediction. Yu and Sturtevant [84] developed the application of artificial intelligence (AI) in two-player fighting games. They discussed the design of these games and how to select the AI types. They provided valuable insights into applying retrograde analysis and Nash equilibria in developing sophisticated AI strategies for fighting games, highlighting AI's potential to compete with human players and enhance the gaming experience. Asayama et al. [85] demonstrated that improving their perception speed enhances AI agents' performance in real-time fighting video games. They developed an AI agent that uses linear extrapolation to predict the opponent's position and the k-nearest neighbor method to predict their actions. They demonstrated that predictive capabilities in AI agents can lead to superior performance compared to non-predictive agents and enable them to compete more effectively against human players. Justesen and Risi [86] demonstrate that deep learning can effectively learn StarCraft's macro management tasks by analyzing highly skilled players' replays. Their approach, which involves training a neural network on state-action pairs from these replays, successfully predicts in-game production decisions, significantly improving the performance of the AI bot. They suggest the potential for deep learning, combined with reinforcement learning, to narrow the performance gap between AI and human players in complex strategy games like StarCraft. Cho et al. [87] proposed a novel approach for enhancing AI bots in StarCraft by utilizing replay data to predict opponent strategies and adapt build orders accordingly. The research demonstrates the potential for AI bots to dynamically adapt strategies in real-time strategy games, a significant advancement from their traditionally rigid behavior.

Our proposed methods draw inspiration from zero-delay and input prediction techniques. These methods are innovative in introducing heuristic pattern reduction and LSTM-based input prediction, enhancing the feasibility of speculative execution. Furthermore, our approach is not confined to specific game genres or operating conditions, assuming various games and operating terminals within a cloud gaming ecosystem.

2.3.3 Speculative Execution on Cloud Gaming

Speculative execution is a pivotal technique in reducing latency in cloud gaming, focusing on predicting potential user actions and pre-rendering corresponding game states. This approach minimizes the delay between user input and the resulting on-screen action. The core of speculative execution lies in developing sophisticated algorithms capable of accurately predicting player actions. These algorithms analyze past player behavior, current game context, and possible future scenarios to forecast the player's next moves. By doing so, the system can pre-render multiple potential outcomes, ensuring that the game state corresponding to the player's choice is ready to be displayed with minimal delay. Anand and Wenren [88] proposed a predictive cloud gaming paradigm, CloudHide, which aims to reduce response latency in cloud gaming by pre-generating and sending future game frames to thin clients in advance. Their approach utilizes the cloud's processing power and network bandwidth to enhance the gaming experience by masking inherent latencies.

On the other hand, the core challenge of speculative execution lies in developing sophisticated algorithms for accurate player action prediction and analyzing past behavior, current context, and future scenarios. Inaccurate predictions pose unique challenges, such as excessive pre-rendering and significant data overhead due to recovery from mispredictions. Frame rate fluctuations due to frequent prediction errors may impair the user's experience more than the delay. Some studies proposed error compensation with video processing. Lee et al. [89] introduced Outatime, a speculative execution system for mobile cloud gaming that effectively masks network latencies up to 250 ms, a threshold critical for maintaining gaming responsiveness. Outatime predicts future user inputs, generates speculative frames of potential outcomes, and employs techniques like state space subsampling, misprediction compensation, and bandwidth compression to provide real-time gaming interactivity. The system's efficacy is demonstrated through user studies and performance benchmarks with commercial games, showing that Outatime significantly improves the gaming experience over high-latency networks while maintaining acceptable visual and interactive quality. Sadaike et al. [90] proposed the enhancement of QoE in cloud gaming by implementing layer caching and motion prediction techniques. The study demonstrates how these methods can significantly reduce network bandwidth usage and response time by employing a Cloud Manager that processes and predicts game scene changes based on player inputs. This approach leads to more efficient

rendering of complex scenes and smoother gameplay, optimizing the use of server resources in cloud gaming environments. They present a theoretical model to analyze this paradigm, demonstrate its potential through a prototype game, and discuss future directions for optimizing computational resources and processing techniques within this framework. Boudaoud et al. [91] explore applying late warp techniques, traditionally used in virtual reality, to reduce latency in First Person Shooter (FPS) games, enhancing player performance. Through an interactive web-based application, they demonstrate how late warp can effectively mitigate the negative impacts of latency on aiming in FPS games. This study highlights the potential of adapting VR technologies to improve user experience in competitive online gaming environments, especially under high latency conditions.

A significant aspect of research in speculative execution is finding the optimal balance between computational load and network bandwidth usage. While pre-rendering game states can reduce latency, it also increases the demand for server resources and network bandwidth. In particular, assuming the most straightforward speculative execution, the server sends all possible frames and exponentially increases traffic. Efficiently managing this trade-off is critical to implementing speculative execution without overwhelming the cloud gaming infrastructure or degrading the quality of service for other users. Future research must solve the trade-off to realize speculative execution in cloud gaming. The key subjects include refining predictive models to enhance accuracy and efficiency. The subject requires improving how these models predict player actions and ensuring they do so with minimal computational overhead and reduced bandwidth usage.

This paper introduces novel ways to reduce the computational resources required for speculative execution more efficiently. Additionally, reducing the bandwidth demands of this process is another critical subject of this research to make speculative execution more feasible and effective for a broader range of network environments.

Chapter 3

Speculative Execution: Pattern Reduction Method

3.1 Introduction

In the rapidly evolving world of digital entertainment, cloud gaming has emerged as a revolutionary technology. Cloud gaming systems free players from hardware constraints by shifting processing loads from local devices to remote cloud servers. This service allows even relatively low-powered devices to deliver high-quality gaming experiences, making cutting-edge games accessible to a broader range of gamers. However, reliance on remote servers in gaming introduces a significant challenge: latency. Latency, the delay between a player's action and the game's response, is particularly critical in cloud gaming. This delay is minimal in traditional gaming setups like consoles or PCs running games locally. However, data must travel back and forth in cloud gaming to distant servers, leading to additional network-induced delays. Zadtootaghaj et al. [92] show the effect of latency for the QoE in cloud gaming systems. The latency can severely impact gameplay, especially in genres that require quick reflexes, like first-person shooters or fast-paced strategy games. On the other hand, even within the same game, there are significant differences in latency sensitivity under different scenarios. Therefore, even in games that are not relatively fast-paced, response delay may be critical.

Some studies have explored speculative execution in cloud gaming systems to fundamentally solve the issue of response latency. Speculative execution involves predicting a player's future input patterns and pre-rendering the corresponding game frames. By sending these pre-rendered frames to the player's device in advance, the aim is to reduce perceived latency. In theory, transmitting all possible input patterns in advance through speculative execution achieves zero latency, resulting in a very high-quality gaming experience. Here, note that speculative execution in cloud gaming systems aims to hide response delays, not for time synchronization in online multiplayer games or compensation for time synchronization. For example, a synchronization system like GGPO [80]

applied to fighting games displays the opponent's continuous input in advance and reduces the overall delay by redrawing with the correct input according to feedback notifications from the server. This idea assumes rollback with feedback, which is different from speculative execution.

Speculative execution poses significant challenges, including increased computational load and network traffic on servers and clients. Predicting and rendering all exponentially increasing potential patterns is practically impossible due to the near-infinite processing power required, and even if feasible, transmitting all these through the network is impractical. Correcting predictions on the client side to consider server processing power and traffic is possible, but demanding high computational power from devices used for cloud gaming is not desirable. Not resolving these potential speculative execution issues could overwhelm both cloud servers and network bandwidth, potentially exacerbating rather than alleviating the problem.

We propose a pattern reduction method in cloud gaming systems to enable efficient speculative execution, addressing previous approaches' inherent computational and network challenges. Our proposal marks a significant consideration towards realizing speculative execution in cloud gaming. The cornerstone of our proposal is a bit-field-based pattern reduction technique. This method narrows down the inputs a player will likely conduct by analyzing the frequency and order of a user's input patterns. Reducing the number of patterns that require speculative processing simultaneously reduces both the computational load and the amount of traffic. Furthermore, we explore using Long Short-Term Memory (LSTM) networks in our pattern prediction strategy. These networks are adept at recognizing and predicting complex patterns in time-series data, making them well-suited for analyzing gaming inputs. LSTM-based pattern prediction enhances the accuracy of speculative execution, ensuring that the high-probable patterns or the most probable pattern of player actions are anticipated and rendered in advance.

The paper's comprehensive evaluation, using data from various game genres, demonstrates the effectiveness of this approach. The proposed methods significantly reduce the number of rendered frames and network traffic compared to previous speculative execution techniques, offering a scalable and effective solution for various gaming genres and titles.

The contributions of this research are as follows:

- Innovative Pattern Reduction Method:

  We introduce a bit-field-based pattern reduction method for speculative execution in cloud gaming, reducing computational load and network traffic.

- Application of LSTM Networks for Input Prediction:

  Using LSTM networks to capture temporal patterns in user input logs enhances the speculative execution process, especially in complex gaming environments.

- Comprehensive Evaluation and Validation:

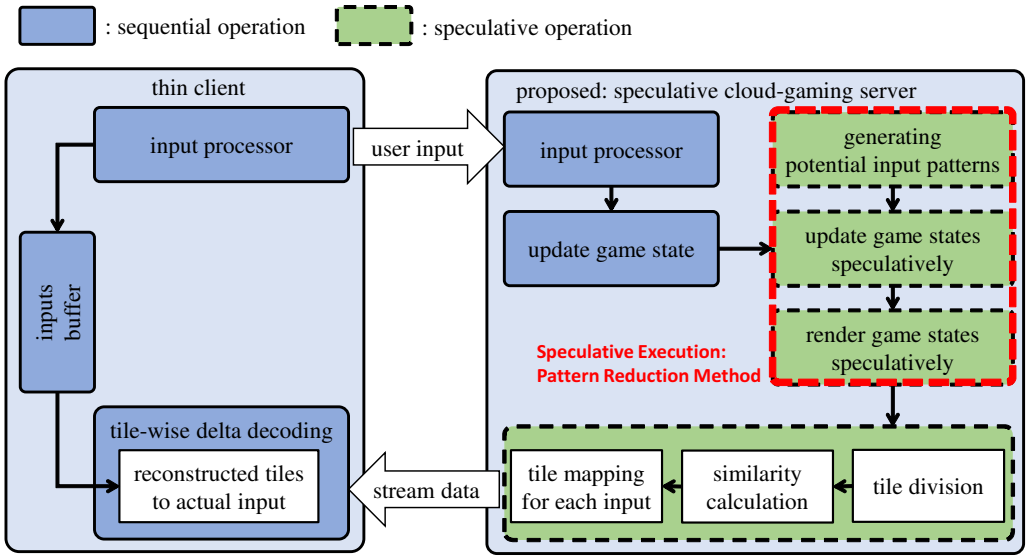


Figure 3.1: Architecture of the Proposed Cloud Gaming System

We extensively tested the method across various game genres, proving its effectiveness and scalability in different gaming contexts.

- Potential for Broad Application in Cloud Gaming:

  Our research suggests a scalable and versatile approach that could be broadly applied in cloud gaming systems, potentially benefiting various games and devices.

- Contribution to Latency Reduction in Cloud Gaming:

  This paper tackles a crucial challenge in cloud gaming — reducing latency and traffic — pivotal for improving the overall user experience in cloud-based gaming services.

3.2 Proposed Method

3.2.1 Overview

Fig. 3.1 shows an overview of the cloud gaming server in the proposed method. First, a user sends inputs over the network to the cloud gaming server. The cloud gaming server collects and analyzes the input logs from the user and generates log-based input patterns for speculative execution. Next, the server updates game states speculatively according to the input the user sends, based on the log-based input patterns generated in advance. Finally, the server renders the game states to video frames and encodes to stream data.

Fig. 3.2 shows the flow of the speculative execution process on a cloud gaming server. The number of possible input patterns in each frame is assumed to be two, and the server performs speculative execution for two future frames. The main thread handles the bit-field data received

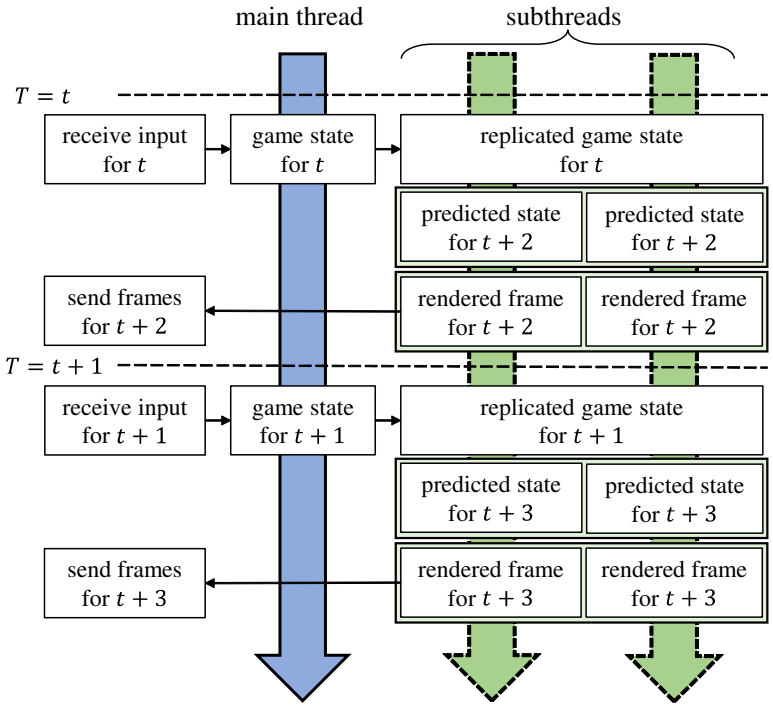


Figure 3.2: Threading process in cloud gaming servers with speculative execution for two future frames. Here, the cloud gaming server has transmitted the user's game frames for $t+1$ at $T = t-1$ and t at $T = t-2$ to the user.

in the cloud server, and several subthreads are used for speculative execution. The cloud gaming server updates the current game state on the main thread based on the received bit-field data. The server then replicates the updated game state to multiple subthreads. Each subthread asynchronously updates the game state up to the following two frames. The corresponding game frame is rendered based on the game's final state on each subthread, and the game frame is encoded and transmitted to the user.

3.2.2 Bit-Field-Based Pattern Reduction

The cloud gaming system corresponds to various combinations of games and operating devices. Therefore, the pattern reduction methods should not be limited to specific games or devices. Our approach enables pattern reduction regardless of the game or device by treating the input signal independently as bit-field data. More specifically, pattern reduction is performed universally by treating the entire data as time-series bit-field data without establishing a correspondence between a specific input signal and each bit string. This section proposes two pattern reduction methods based on bit-field time-series data: the temporal pattern analysis (TPA) method and the LSTM-based pattern prediction (LBPP) method.

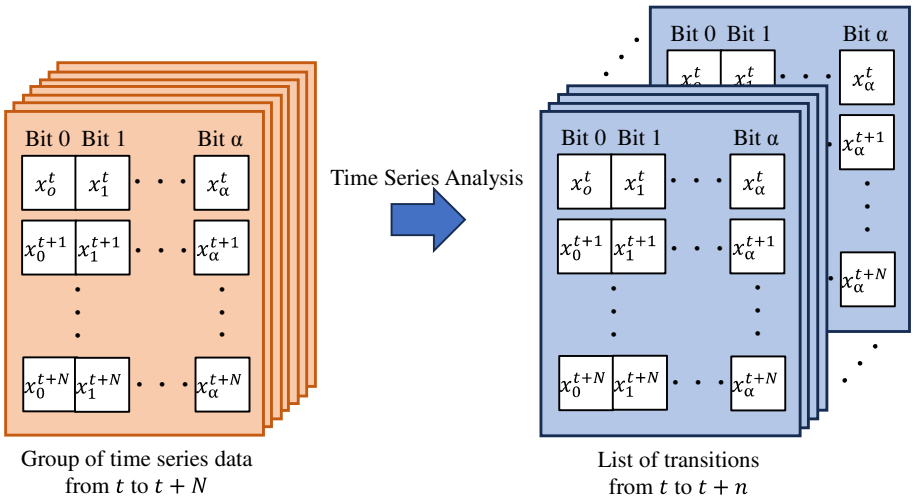


Figure 3.3: Overview of the TPA

Temporal Pattern Analysis Method

Fig. 3.3 shows the overview of the TPA method. The TPA works by detecting frequent transitions in the input logs over time. More specifically, the server creates a list of patterns through time-series analysis and determines speculative execution patterns from that list. From the input log for period N obtained from the user, the pattern detection procedure obtains and lists the transitions in period n from $t \rightarrow t+n$ and calculates the probability that occurs. Here, x_i^t is a binary value of 0 or 1 that indicates the value of each bit. Therefore, if the bit length is α and the pattern length is n, processing a maximum of $2^{\alpha n}$ patterns is necessary. Each transition is associated with a specific probability, labeled as $p_{0,n}$, $p_{1,n}$, up to $p_{2^{\alpha n},n}$. In the proposed system, a transition with a probability more significant than a threshold is selected from a list, and speculative execution is performed. Besides, the computational space will be manageable because it can be analyzed after extracting a unique bit pattern. Additionally, by constructing a pattern tree structure in advance, the search speed during actual execution can be completed in a few milliseconds.

Fig. 3.4 shows the TPA procedure. Here, the length of the bit-field data is assumed to be four. The cloud gaming server analyses a time series of bit-field data for previous users and uses unique bit-field data listed from all previous users' logs. Redundant input patterns other users have never entered can be removed from speculative execution. The cloud gaming server selects each unique bit-field data set and counts the number of transitions from the selected bit-field data to others based on user logs. Redundant transitions that other users have never entered can be removed in the time domain. Note that the cloud gaming server can set the priority of the transitions based on the number of transitions. Finally, the tree structure for each unique bit-field data is constructed based on the above operations, and the bit-field data corresponding to the leaf nodes are used for

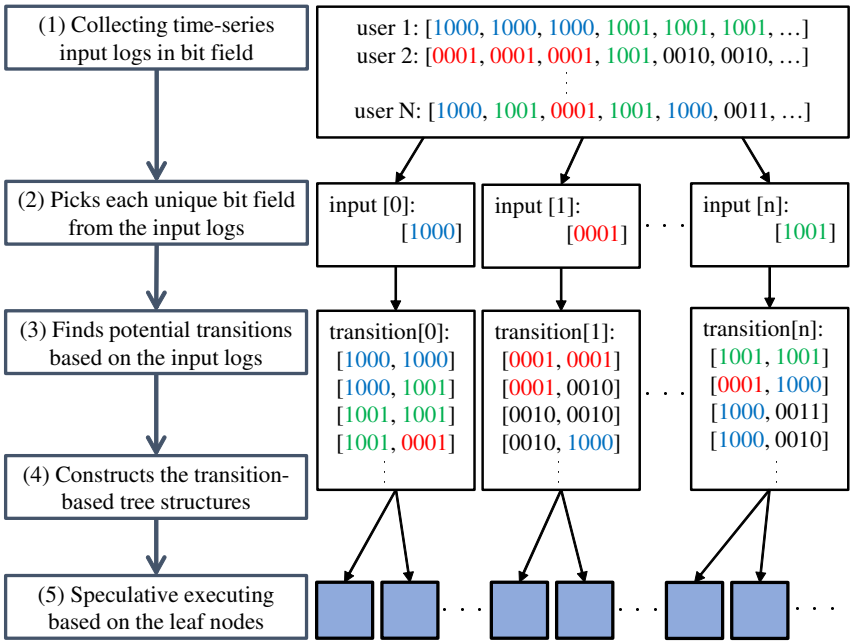


Figure 3.4: Proposed bit-field-based TPA. In this case, the analysis is carried out for two future frames.

speculative execution.

Here, we define the average framerate and the number of patterns per frame in the pattern detection procedure. First, if the bit length is α and the pattern length is n, the sum of the probabilities is 1.

$$\sum_{i=0}^{2^{n\alpha}} p_{i,n} = 1 \tag{3.1}$$

When transitions above the threshold τ are used for speculative execution, the probability of successful prediction is as follows:

$$P_{n,\tau} = \sum_{i=0}^{2^{n\alpha}} p_{i,n}^{[i|0\leq i\leq 2^{n\alpha}, p_{i,n}\geq\tau]} \tag{3.2}$$

Therefore, the average framerate is determined as follows:

$$R_{n,\tau} = r \cdot P_{n,\tau} \tag{3.3}$$

Also, since only transitions that are equal to or greater than the threshold are processed, the number of speculative processing per frame is determined as follows:

$$F_{n,\tau} = |\{i|0 \leq i \leq 2^{n\alpha}, p_{i,n} \geq \tau\}| \tag{3.4}$$

LSTM-based Pattern Prediction Method

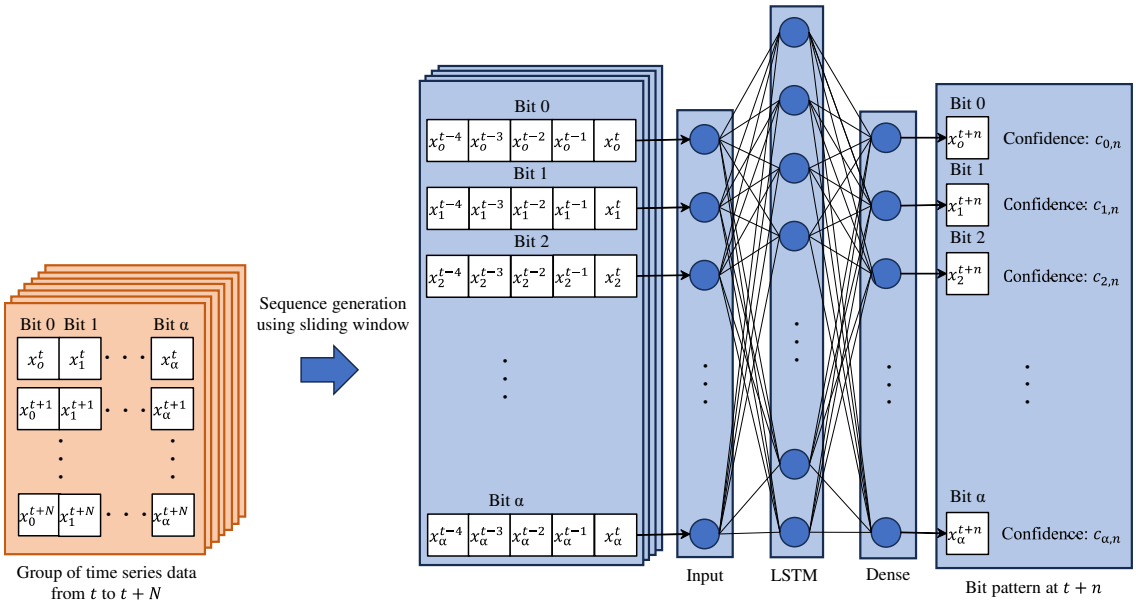


Figure 3.5: The overview of LBPP

Fig. 3.5 shows the overview of the LBPP method. The LBPP uses LSTM neural networks to model short and long-term temporal patterns in the input logs. It then predicts input probabilities to estimate likely future inputs for speculative execution. In the proposed system, we consider generating the most reliable pattern and multiple patterns based on the confidence level of each bit. For example, when building a prediction model for n frames ahead, $N - n + 1$ sequences are generated from the log of period N based on a sliding window of period n. Input the generated sequence into the LSTM layer of the neural network. The LSTM layer analyzes the time dependence of the sequence data and inputs the results as features to the Dense layer. Finally, for each bit $\{i|0 \leq i \leq \alpha\}$, the model outputs the confidence $c_{i,n}$ that the bit will be 0 after n frames. More specifically, bits with confidence greater than or equal to the threshold τ are considered definite, and bits with confidence less than the threshold τ are considered undetermined. This approach improves the prediction n frames ahead and increases the success probability of speculative processing.

The algorithm 1 shows the algorithm of LBPP. LBPP processes the following steps:

1. Initialize data: An empty list named 'data' is initialized to hold the dataset.

2. Read data from files: Data is read from each file and added to the 'data' list after undergoing a transformation process to format it for the LSTM model.

3. Data preprocessing: Input sequences and corresponding labels are prepared from the 'data' list, which is then used to train the LSTM model.

4. Training LSTM model: The LSTM model is trained using the preprocessed data, learning to predict the output labels from the input sequences.

Algorithm 1 The algorithm of the LBPP

```
Initialize empty list 'data'
for each file do
    Append transformed data from file to 'data'
end for
Prepare training data and labels
Train the LSTM model

Function SLIDING_WINDOW(arr, window_size, start)
return arr[start : start + window_size]

Function EVALUATE_PREDICTION(model, input_seq, steps)
for each step do
    window = SLIDING_WINDOW(data, seq_length, current)
    Predict the next data point using model and window
    Compute metrics (average probabilities, time taken, etc.)
    Record the metrics
    Update current
end for
return predictions

Function REALTIME_PREDICTION(model, data_stream, N)
while new data is available from data_stream do
    input_seq = Receive new data point at time t from data_stream
    Initialize empty list for predictions predictions
    for n = 0 to N − 1 do
        window = SLIDING_WINDOW(input_seq, seq_length, t + n)
        prediction = model.predict(window)
        Append prediction to predictions
        Update input_seq with prediction for next step
    end for
    Execute speculative processings based on predictions
end while
```

5. Prediction of steps: Two procedures are introduced for evaluation and real-time prediction.

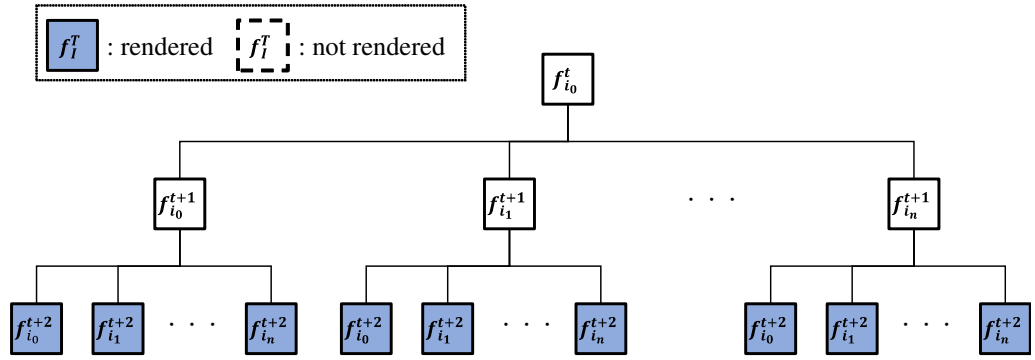
   - Evaluate prediction: This function uses the trained model to predict future data points in a sequence for a given number of steps intended for performance evaluation.
   - Real-time prediction: This function applies the trained model to make real-time predictions for a specified number of future steps.

The confidence that all bits are the following equation represents 0:

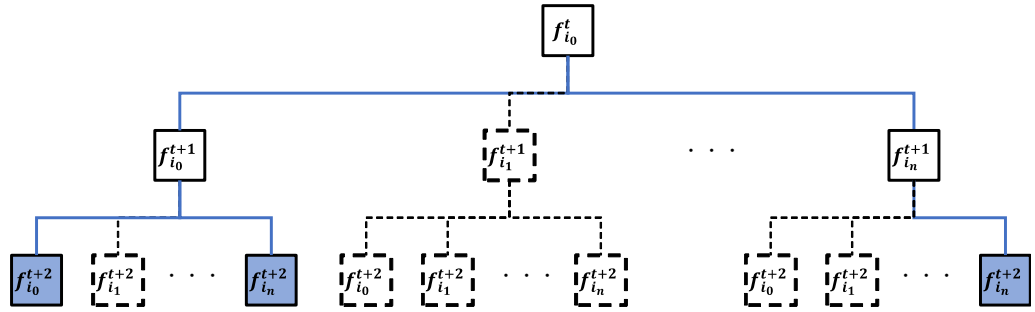
$$\prod_{i=0}^{\alpha} c_{i,n} \tag{3.5}$$

Given a bit number set U, the sum of confidences equals one:

$$\sum_{U \subseteq S} \prod_{i=0}^{\alpha} (c_{i,n}^{[i \in U]} (1 - c_{i,n})^{[i \notin U]}) = 1 \tag{3.6}$$



(a) Speculative execution without input pattern reduction



(b) Speculative execution using proposed bit-field-based temporal pattern analysis

Figure 3.6: The number of game frames rendered and transmitted in the existing and proposed methods. The possible input patterns in each frame are N, and speculative execution is used for two future frames.

Here, $S = \{i|0 \leq i \leq \alpha\}$. For a bit where $x_i^{t+n} = 0$ and its confidence is greater than $x_i^{t+n} = 1$, the set of bit numbersS_n^0 is determined as follows:

$$S_n^0 = \{i|0 \leq i \leq \alpha, c_{i,n} \geq (1 - c_{i,n})\} \tag{3.7}$$

Furthermore, the set of bits for which $x_i^{t+n} = 0$ and the confidence is greater than or equal to ρ is:

$$S_{n,\rho}^0 = \left\{i|i \in S_n^0, c_{i,n} \geq \rho\right\} \tag{3.8}$$

Additionally, the set of bits where $x_i^{t+n} = 1$ and the confidence is greater than or equal to ρ is:

$$S_{n,\rho}^1 = \left\{i|i \notin S_n^0, (1 - c_{i,n}) \geq \rho\right\} \tag{3.9}$$

The total confidence $C_{n,\rho}$ is the sum of the confidences that are greater than or equal to ρ from the sets described by Eq. (3.8), (3.9):

$$C_{n,\rho} = \sum_{i=0}^{\alpha} \left(c_{i,n}^{[i \in S_{n,\rho}^0]} (1 - c_{i,n})^{[i \in S_{n,\rho}^1]} \right) \tag{3.10}$$

In the proposed system, the average framerate $R_{n,\rho}$ is calculated by multiplying the framerate r of the game executed on the server side by the total confidence $C_{n,\rho}$:

$$R_{n,\rho} = r \cdot C_{n,\rho} \tag{3.11}$$

Table 3.1: PC specifications

	configuration
OS	Windows 11
CPU	Intel Core i9-10850K
RAM	128 GB
GPU	NVIDIA GeForce RTX 3080

Furthermore, the number of undetermined bits, represented as $S'_{n,\rho}$, is the set of bits numbers that are not in either $S^0_{n,\rho}$ or $S^1_{n,\rho}$. Thus, the number of patterns per frame, $F_{n,\alpha}$ is determined as follows:

$$F_{n,\rho} = 2^{|S'_{n,\rho}|} \tag{3.12}$$

3.2.3 Speculative Rendering

Fig. 3.6(a) shows an example of the existing speculative execution for two future frames. Here, the number of input patterns at each frame is considered N, and f_i^t represents the game frame at time t corresponding to input i. In this case, the existing method must render and transmit N^2 game frames for speculative execution. Fig. 3.6(b) illustrates speculative execution using the proposed bit-field-based temporal pattern analysis. In this scenario, the cloud gaming server does not find input i_1 and transitions of $i_0 \rightarrow i_0 \rightarrow i_1$, $i_0 \rightarrow i_2 \rightarrow i_0$, $i_0 \rightarrow i_2 \rightarrow i_1$ in the logs of past users. Consequently, the cloud game server will not render the game frames corresponding to the input and transitions.

3.3 Evaluation

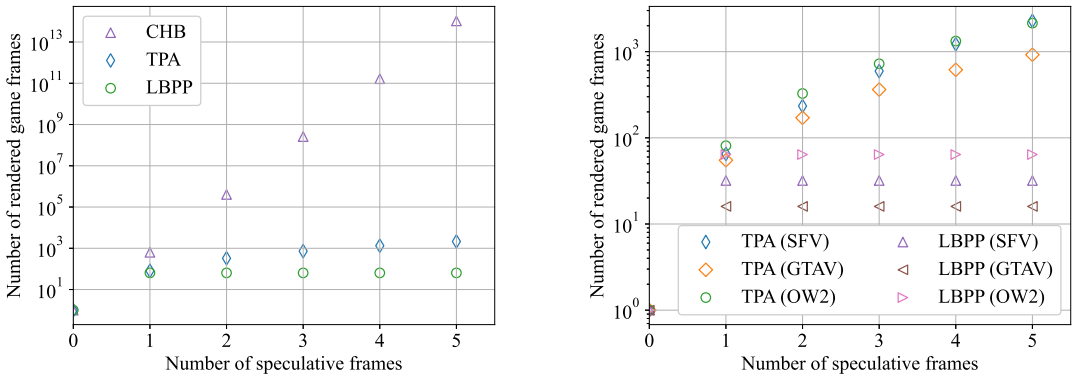
3.3.1 Experimental Measurements

We conducted experimental measurements on users' input logs to assess the efficacy of the proposed method. Ten male subjects, aged 20 to 30, participated in this experiment. Each subject played *Street Fighter V* (SFV), *Grand Theft Auto V* (GTAV), and *Overwatch 2* (OW2) sequentially on a PC using an Xbox One controller. Table 3.1 shows the specifics of the PC. The subjects engaged in the training modes of SFV and OW2 and the early stages of GTAV's story mode. All games ran at 1080p and 60fps, with input logs collected at the game's framerate.

The proposed bit-field-based temporal pattern analysis was implemented in Python 3.8. We used each subject's 10-minute input log to build the tree structures and the remaining 1-minute log for evaluation.

3.3.2 Effect on the Number of Game Frames

We first established baseline performance for the proposed method, comparing it with the CloudHide-based (CHB) method. CHB renders and sends game frames for every possible input pattern.



(a) Effect in OW2

(b) Effect in all games

Figure 3.7: Number of rendered game frames for speculative frames. The average framerate is set to over 57 fps for all methods. (a) compares CHB and the proposed method in OW2. (b) compares the proposed method across all games.

Figure 3.7(a) illustrates the number of speculative frames in OW2, maintaining an average framerate of over 57fps. The results indicate that both proposed methods, TPA and LBPP, effectively suppress the exponential increase in speculative processes for multiple frames, with LBPP showing a near-constant number of frames.

Figure 3.7(b) displays the number of speculative frames in all three games. Despite the varying nature of the games, a similar trend was observed, with the number of patterns consistent across different titles. The result demonstrates the adaptability of the proposed methods to various game genres.

In summary, the key findings are:

- TPA and LBPP significantly reduced the number of speculative execution patterns compared to CHB.
- LBPP maintained a near-constant number of patterns as frames increased.
- The effectiveness of the proposed methods is consistent across different game titles.

3.3.3 Effect of Pattern Reduction Methods

Figures 3.8–3.10 illustrates traffic variation as a function of the required framerate for 1 frame speculative execution. The server-side game runs at a maximum of 60 fps. The CHB method, which generates all possible frames, adjusts the server's operating framerate to achieve the system's required rate. In contrast, our proposed method operates at a fixed framerate of 60 fps. We established a threshold to minimize traffic while maintaining the required framerate.

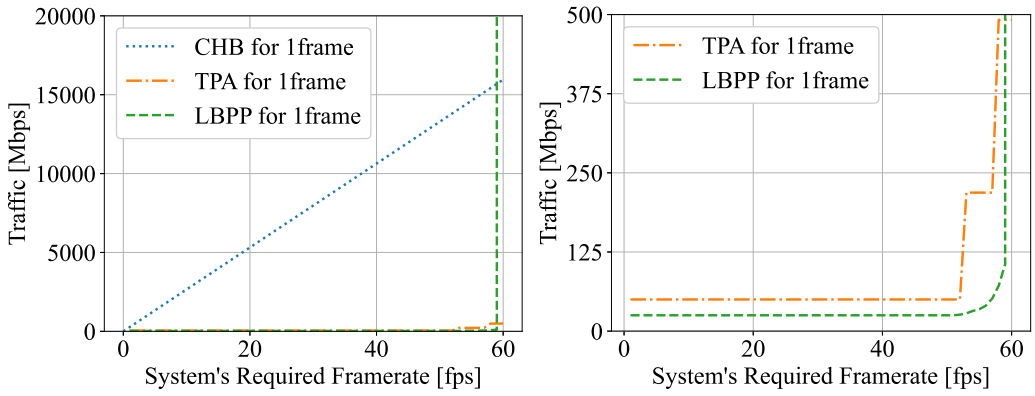


Figure 3.8: SFV: A comparative analysis between CHB and the proposed methods (a), alongside A comparative analysis between the proposed methods (b).

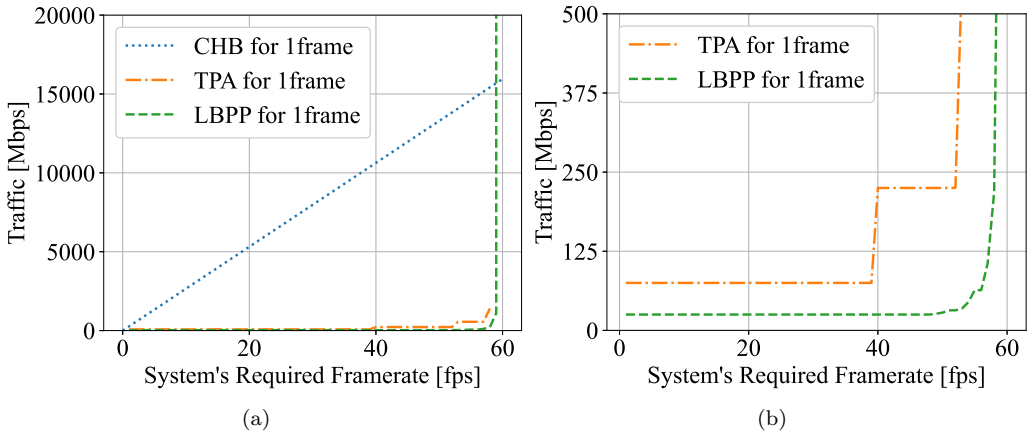


Figure 3.9: OW2: A comparative analysis between CHB and the proposed methods (a), alongside A comparative analysis between the proposed methods (b).

Figures 3.8(a), 3.9(a), and 3.10(a) compare the CHB method with our proposed TPA and LBPP methods. CHB's traffic changes linearly with the required framerate. In contrast, our methods significantly reduce traffic when the required rate is around 57 fps. However, as the required framerate exceeds 58 fps, approaching the server's operational rate, traffic in our proposed methods, especially LBPP, increases sharply. LBPP, based on recurrent neural networks (RNN), does not guarantee 100 % bit reliability, necessitating the transmission of more frames as the target rate nears the server's 60 fps.

Figures 3.8(b), 3.9(b), and 3.10(b) show results with traffic under 500 Mbps. For example, in Figure 3.9(b), TPA maintains about 300 Mbps while operating at 60 fps. TPA can achieve high framerates with low traffic in games with limited input patterns. Conversely, LBPP consistently requires less traffic than TPA for the same framerate across different games, offering stable

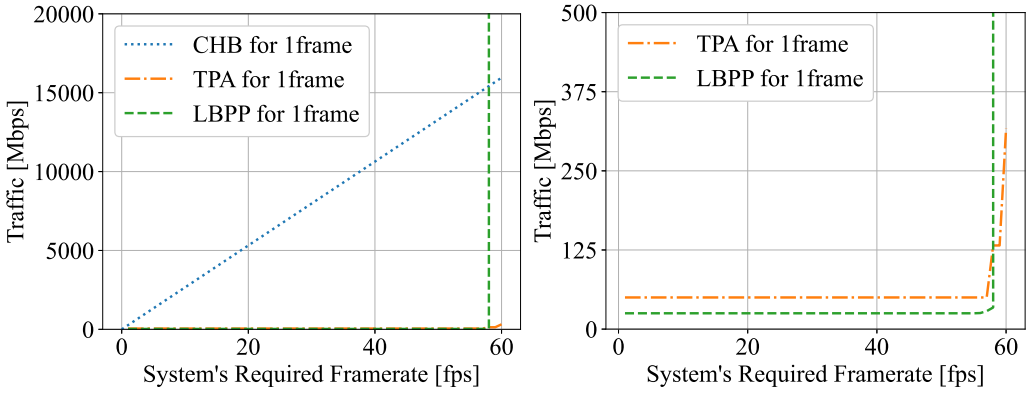


Figure 3.10: GTA: A comparative analysis between CHB and the proposed methods (a), alongside A comparative analysis between the proposed methods (b).

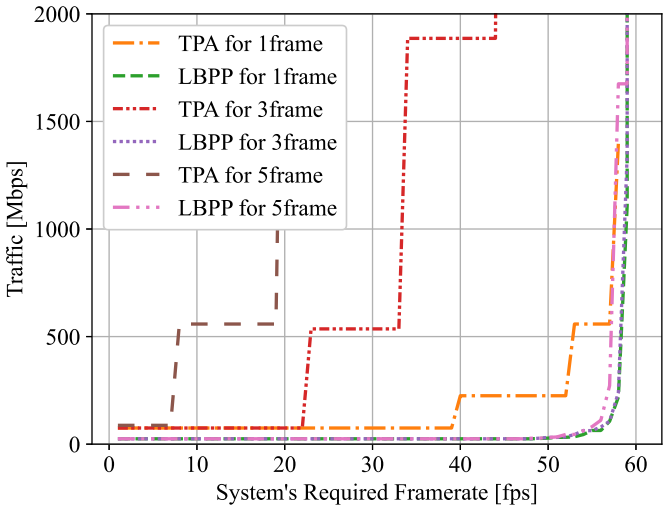


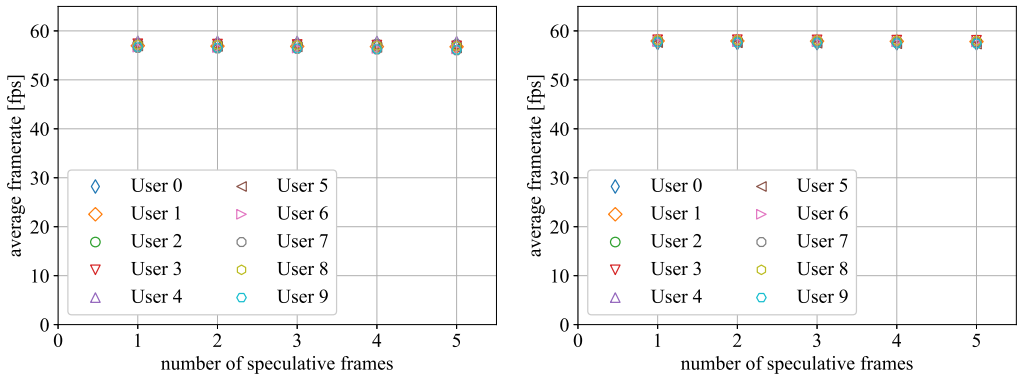
Figure 3.11: A comparison of the proposed methods with 1, 3, and 5 frames speculatively executed in OW2.

framerates and traffic reduction regardless of the game genre.

Figure 3.11 compares the proposed methods in OW2 with 1, 3, and 5 frames speculatively executed. LBPP is notably resilient to increases in speculative frames, maintaining prediction accuracy and pattern consistency. Conversely, TPA's traffic increases to maintain the required framerate, particularly in fast-paced games like OW2.

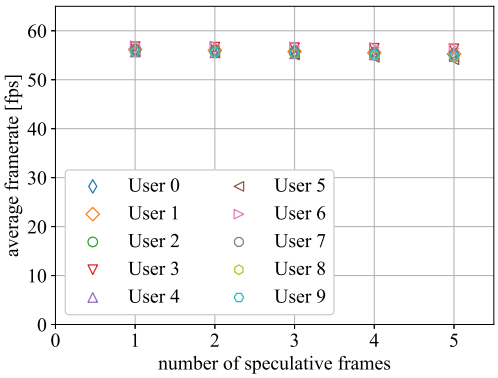
In summary:

- The proposed methods, particularly at around 57 fps, substantially reduce traffic compared to CHB.
- LBPP, being an RNN-based method, requires more frames for higher target framerates, as



(a) SFV

(b) OW2



(c) GTA

Figure 3.12: The average frame rate when the same LBPP-trained model is applied to each user.

bit reliability is not absolute.

- TPA excels in certain genres, but LBPP consistently provides stable framerate and traffic reduction across various game types and is less affected by the number of speculative frames.

3.3.4 Discussion about Trained Models

Figure 3.12(a)–(c) shows the average frame rate when the same LBPP-trained model is applied to each user. This result implies that differences between users do not greatly affect the results in this limited sample group. The learning effect in the same game may be independent of the players' personal characteristics. It should be noted here that the experiment in this study was limited to relatively young men in their 20s in the laboratory. These results are limited in their generality as they do not sufficiently include levels of gaming skills, different age groups, and various game scenes. In this experiment, multiple users were playing the same game scene, so the operations likely were the same. The independence from user characteristics needs to be confirmed through

Algorithm 2 Conversion of QoE expressed in R scale to MOS scale

Require: R_QoE
Ensure: MOS_QoE
$MOS\_MAX \Leftarrow 4.64$
$MOS\_MIN \Leftarrow 1.3$
if $R\_QoE > 0$ & $R\_QoE < 100$ **then**
  $MOS\_QoE = 1 + (MOS\_MAX - MOS\_MIN)/100 \times R\_QoE + R\_QoE \times (R\_QoE - 60) \times (100 - R\_QoE) \times 7.0E - 6$
else if $R\_QoE \geq 100$ **then**
  $MOS\_QoE = MOS\_MAX$
else
  $MOS\_QoE = MOS\_MIN$
end if

Table 3.2: The default constant coefficients of Eq. (3.13)

Coefficient	a	b	c	d	e
Value	0.788	0.896	0.227	0.625	0.848

Table 3.3: Probabilities

frame-num	1	2	3	4	5
SFV	0.416	0.393	0.375	0.367	0.369
GTAV	0.625	0.619	0.614	0.610	0.607
OW2	0.956	0.954	0.951	0.948	0.946

more extensive experiments.

3.3.5 Experimental QoE Evaluation

Reference QoE Model

In our pursuit of QoE evaluation for cloud gaming systems, we introduce a QoE model defined by the International Telecommunication Union in G.1072 [93]. This model quantifies the game experience quality under varying video, audio, and delay conditions. The QoE model for cloud gaming is represented as follows:

$$R_{\mathrm{QoE}} = R_{\max} - a \cdot I_{\mathrm{VQ_{cod}}} - b \cdot I_{\mathrm{VQ_{trans}}} - c \cdot I_{\mathrm{TVQ}} - d \cdot I_{\mathrm{IPQ_{frames}}} - e \cdot I_{\mathrm{IPQ_{delay}}} \tag{3.13}$$

Here, R_{QoE} predicts the game's QoE on an R-scale, ranging from 0 (worst quality) to 100 (best quality). The reference value, $R_{\max}$, is set at 100. Five components—$I_{\mathrm{VQ_{cod}}}, I_{\mathrm{VQ_{trans}}}, I_{\mathrm{TVQ}}, I_{\mathrm{IPQ_{frames}}}$, and $I_{\mathrm{IPQ_{delay}}}$—are calculated in R-scale to determine the QoE. These five predicted values, each weighted by coefficients a, b, c, d, and e, are subtracted from $R_{\max}$ to predict R_{QoE}. Thus, lower predicted values indicate higher QoE.

Our predictions incorporate latency and packet loss as network parameters and frame resolution, encoded frame rate, and bit rate as coding parameters. $I_{\mathrm{VQ_{cod}}}$ gauges game video quality, factoring in frame resolution, encoded frame rate, and bit rate. Higher game image quality on the client side lowers the $I_{\mathrm{VQ_{cod}}}$ value. $I_{\mathrm{VQ_{trans}}}$ assesses game video quality, considering packet loss and $I_{\mathrm{VQ_{cod}}}$. Improved network quality between server and client reduces $I_{\mathrm{VQ_{trans}}}$. I_{TVQ} measures temporal

video quality, focusing on frame loss due to subpar network conditions. When the client-side average frame rate matches the server-side encoded frame rate, I_{TVQ} is minimized. $I_{\mathrm{IPQ}_{\mathrm{frames}}}$ evaluates input quality, accounting for frame loss on the client side. An average frame rate closer to the maximum on the client side decreases $I_{\mathrm{IPQ}_{\mathrm{frames}}}$. $I_{\mathrm{IPQ}_{\mathrm{delay}}}$ quantifies input quality using network delay. Reduced network delay between server and client lowers $I_{\mathrm{IPQ}_{\mathrm{delay}}}$. We convert the predicted R_{QoE} into a Mean Opinion Score (MOS), denoted as MOS_{QoE}. Here, scores from 1 to 5 represent bad, poor, fair, reasonable, and excellent quality, respectively. In our study, MOS_{QoE} ranges from 1.3 (worst quality) to 4.64 (best quality).

To demonstrate the impact of our method on the QoE model, we analyze the parameters in Eq. (3.13). We can categorize these parameters into those that affect the quality of the video and input experience. Parameters a, b, c, and d influence video experience quality, while e impacts input experience quality. The weight of these parameters varies depending on the game's characteristics.

QoE Model Optimization

We further refine the QoE model to account for the impact of our proposed method. We assume $\alpha = \frac{a+b+c+d+e}{a+b+c+d+W \cdot e}$, where W is the weight for input experience quality. We then modify Eq.3.13 as follows:

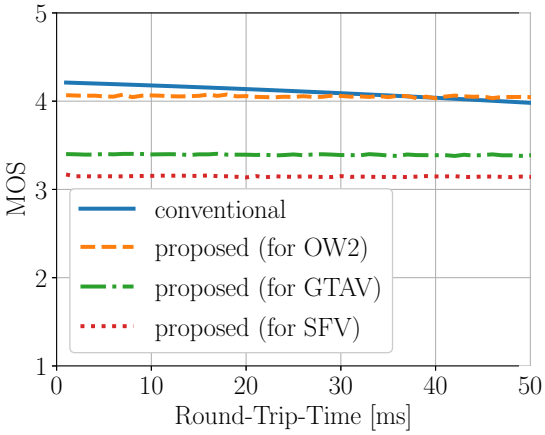
$$R_{\mathrm{QoE}}^{opt} = R\max - \alpha(a \cdot I_{\mathrm{VQ}_{\mathrm{cod}}} + b \cdot I_{\mathrm{VQ}_{\mathrm{trans}}} + c \cdot I_{\mathrm{TVQ}} + d \cdot I_{\mathrm{IPQ}_{\mathrm{frames}}} + W \cdot e \cdot I_{\mathrm{IPQ}_{\mathrm{delay}}}) \quad (3.14)$$

An increase in the W value elevates the importance of input experience quality. The values for a, b, c, d, and e are set at 0.788, 0.896, 0.227, 0.625, and 0.848, respectively. When W is 1, 3, and 6, the weights of the input experience quality account for approximately 25 %, 50 %, and 75 % of QoE, respectively. We estimate the QoE based on the Round Trip Time (RTT) using Eq.(3.14). It is important to note that without our proposed method, the estimated QoE would be uniform across all games as game characteristics are not considered in Eq. (3.14).

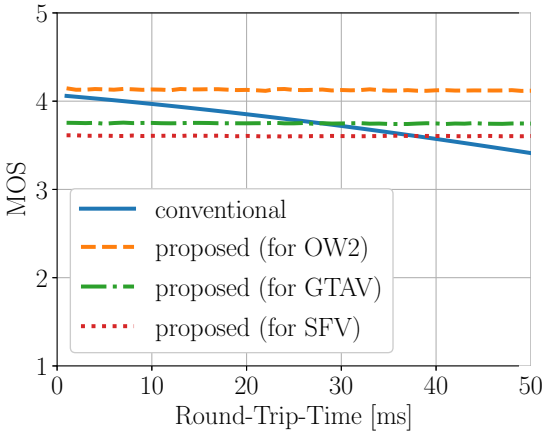
QoE Evaluation

Additionally, we conducted an experimental QoE evaluation using user input logs to empirically assess our proposed method's effectiveness. We utilized a simple one-layer LSTM model on these logs to calculate the average of the highest transition probabilities between frames, namely, the highest confidence in Figure 3.5. We can deploy the speculative cloud gaming system that uses only one pattern with the highest probability without changing the current game engine. Table 3.3 presents the experimentally determined averages of the highest transition probabilities. We collected input logs in specific game modes: training in SFV, early stages of story mode in GTAV, and training in OW2.

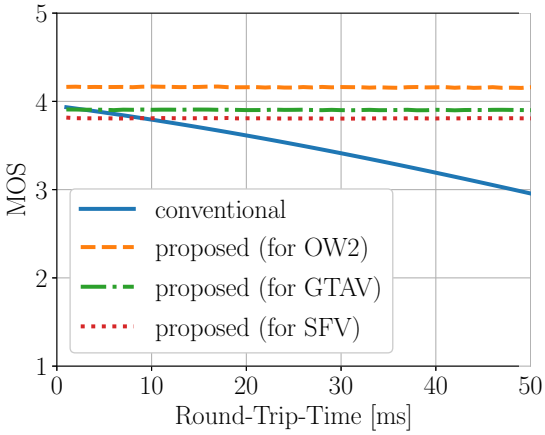
To demonstrate the impact of our proposed cloud gaming system, we compared it with a conventional system using the optimized QoE model in Eq.(3.14). We evaluated our method



(a) Input experience quality constitutes approximately 25% of QoE MoS (W=1)



(b) Input experience quality constitutes approximately 50% of QoE MoS (W=3)



(c) Input experience quality constitutes approximately 75% of QoE MoS (W=6)

Figure 3.13: Composition ratio of input experience quality in QoE MoS

by varying the relative importance of video and input experience quality. We set parameters including a maximum frame rate of 60 fps, a bit rate of 30 Mbps, an RTT of 50 ms, and a game processing delay of 32 ms. The average frame rate was determined by multiplying the prediction accuracy from Table3.3. For comparison, we used a conventional cloud gaming system based on a request-and-response method. In this evaluation, our proposed system employs a pre-transmission method, predicts several frames based on the response delay, generates the most probable frame, and transmits it in advance. The frame appears without delay when the prediction is accurate from the client's perspective.

Figures 3.13(a)–(c) depict the estimated QoE comparison based on RTT for weightings of $W = 1, 3$, and 6. With $W = 1$, prioritizing video quality, the QoE remains stable even in the traditional system, and our method shows a minor effect. Conversely, with $W = 3$, where delay

and video quality are equally important, the QoE drops below four with delays over six milliseconds in the conventional system. In action games with RTT over 30 ms and fighting games with RTT over 38 ms, our system achieves higher estimated QoE. At $W = 6$, where latency outweighs video quality, the estimated QoE in the conventional system never surpasses four.

Although this evaluation uses a general-purpose QoE estimation model that supports multiple game genres, some studies have suggested that frame rate drops are unimportant in certain genres. Schmidt et al. [94] conducted a QoE evaluation in GTAV, pointing out no significant difference in QoE between 25 and 60 fps. In games where the frame rate is not important, even when speculative execution is performed using only maximum likelihood patterns, the effect of improving the quality of experience due to delay reduction may be emphasized.

Key observations from our study include:

- Our system enhances QoE in environments with high RTT.
- In FPS games, QoE improvement occurs with weightings of $W = 3$ and $W = 6$, regardless of RTT.
- A small decrease in prediction accuracy allows our system to mitigate QoE effects even with extended propagation delays.

3.4 Conclusion

This chapter introduces a novel pattern reduction method employing bit-field representations to facilitate efficient speculative execution in cloud gaming systems. The proposed method utilizes TPA and LBPP to analyze user input logs to reduce input patterns that should be processed speculatively. Experimental results obtained from operation logs across multiple game genres by multiple users show that our approach effectively reduces the number of patterns processed in speculative execution across various game titles used in cloud gaming. The key insights from our study are as follows:

1. Our pattern reduction methods effectively curb the exponential increase in speculative patterns across multiple frames.
2. The LBPP method consistently maintains stable frame rates and achieves traffic reduction across different game genres.
3. The TPA method exhibits strong performance in certain genres, closely aligning system frame rates with those on the server, albeit within certain limits.

Furthermore, we explored the effectiveness of pre-sending game frames for inputs with the highest predicted probability. Speculative execution of one pattern with the highest probability

is possible without changing the current game engine. We assessed the QoE by contrasting our method with a conventional cloud gaming system, employing an adapted QoE model. Our proposed method significantly enhances system responsiveness while potentially reducing the average frame rate in cases of prediction errors. Consequently, our system can boost QoE, especially when response delay holds greater significance than video quality.

Our approach demonstrated scalability across various games, independent of specific titles or input devices. Future research will concentrate on refining prediction methodologies. We aim to elevate prediction accuracy and introduce a dynamic mechanism for selecting multiple prediction methods, balancing performance and execution time. This pattern reduction strategy has unveiled the potential to minimize computational demands to levels conducive to the practical implementation of speculative execution, effectively concealing network delays and enriching the cloud gaming experience.

Chapter 4

Speculative Execution: Dedicated Video Transmission Method

4.1 Introduction

As cloud gaming services become more popular, addressing the issue of response delays is becoming more critical. Chapter 3 focused on enabling speculative execution. Despite achieving speculative execution within an acceptable computational cost, the results in Chapter 3 indicate that the volume of generated video data can increase several to dozens of times. This realization of speculative execution brings new burdens on network traffic and server processing capabilities.

Therefore, to practically implement speculative execution in cloud gaming systems, it is necessary to consider traffic reduction. Based on differential coding, existing cloud gaming systems employ video compression techniques like H.264 or HEVC. However, these differential coding techniques cannot be applied directly in cloud gaming systems using speculative methods. Since speculative processing requires sequential processing of each frame, the correct frame from the user's perspective is not known on the server side at the time of transmission, preventing the calculation of differences. Adopting a method that effectively reduces traffic within multiple frames co-occurring is essential for cloud gaming systems utilizing speculative methods.

This research proposes a method to effectively control the increase in traffic while enabling practical implementation of speculative execution in cloud gaming. The proposed traffic reduction method identifies and transmits only the changed parts between a selected reference frame and other generated frames. By significantly reducing the amount of video data required, this method eases the network burden and lowers server processing loads. Our approach divides all frames into multiple tiles, using the frame most likely to represent the user's input as the reference frame. Using a hash function, calculate the similarity between tiles in the reference frame and tiles in other candidate frames. Regarding candidate frames, we send only tiles dissimilar to tiles in the reference frame. This tile-wise delta detection method aims to maintain video quality while reducing traffic

volume under the unique environment of speculative execution, where differential coding techniques are not applicable. It offers an innovative approach to address the trade-off between hiding latency and managing traffic in speculative execution.

The contributions of this research are as follows:

- Introduction of an Innovative Tile-wise Delta Detection Method:

  We present a new approach to reduce traffic in speculative video transmission in cloud gaming by selectively transmitting only the altered parts of a video frame.

- Reduction in Network Traffic:

  This method significantly decreases the volume of video data transmitted, which is essential for managing network bandwidth and enhancing the efficiency of cloud gaming systems.

- Maintaining Video Quality:

  Despite the reduced traffic, our method maintains a high level of video quality, which is crucial for an immersive gaming experience.

- Broad Applicability Across Game Genres:

  The effectiveness of this method is demonstrated across various games, suggesting its versatility and broad applicability in the cloud gaming industry.

- Practical Feasibility in Current Cloud Gaming Infrastructure:

  We explore the feasibility of implementing this method using current computational capabilities, emphasizing its practical applicability in existing cloud gaming infrastructures.

4.2 Proposed Method

4.2.1 Overview

In this section, we propose a method to reduce traffic in a cloud gaming system implementing speculative execution. Fig. 4.1 shows the overall structure of our proposed cloud gaming system. This architecture introduces additional functions to existing cloud gaming systems, enhancing the efficiency of speculative execution. The cloud gaming server generates all possible input patterns for each user and reduces the number of potential input patterns based on the methods described in Chapter 3. Then, the system renders video frames based on these input patterns and the current game state. Each rendered video frame is divided into multiple tiles. A reference frame is selected using a hash function, and the similarity with other video frames is calculated. Tiles with high similarity are not transmitted, while those with low similarity are encoded and sent to the user. Users combine the tiles from the reference video frame with the received ones to decode the video

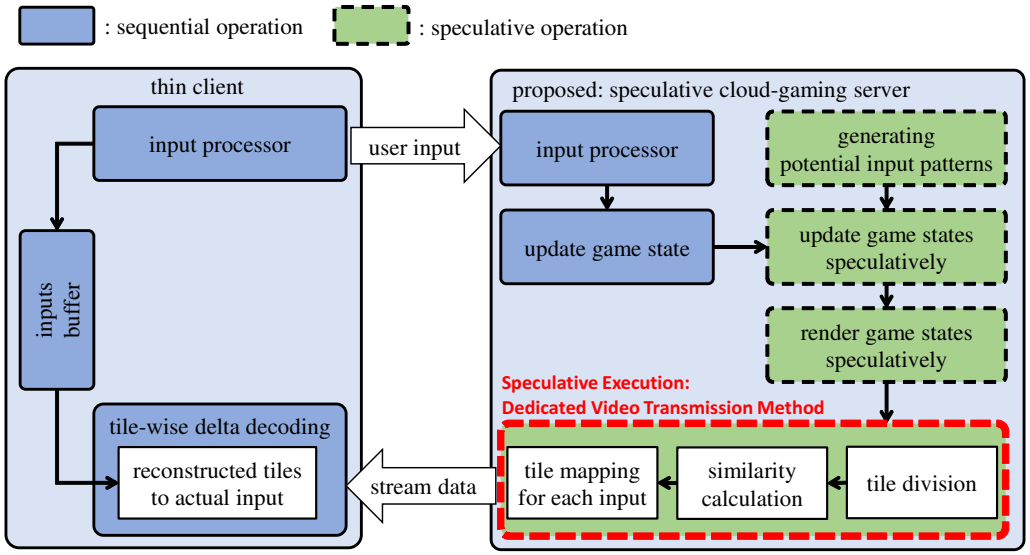


Figure 4.1: Architecture of the Proposed Cloud Gaming System

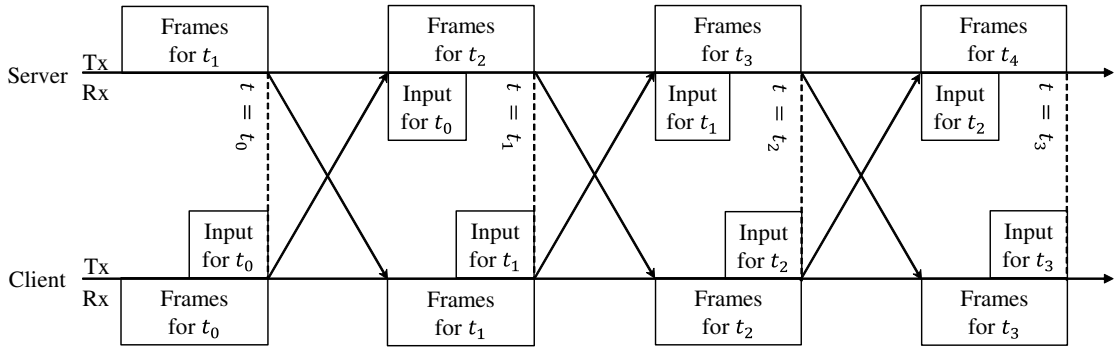


Figure 4.2: Data Flow between Server and Client

frame corresponding to the actual input and display it. This process reduces the increased traffic due to speculative execution while masking network latency.

Fig. 4.2 presents a time-sequence chart of the data flow between the cloud gaming server and the client. This chart illustrates the synchronized data exchange between the server and client under ideal conditions, clarifying the timing and process of the speculative method. For example, based on network latency and frame rate, we set the number of speculative frames to 2. The server updates the game state based on the user's input and prepares the results for time t_1. This interaction is repeated for each frame, offering an optimal experience to the user.

4.2.2 Tile-wise Delta Detection

Fig. 4.3 presents an overview of our proposed tile-wise delta detection method. In this method, the cloud gaming server generates video frames based on users' potential input patterns. It mitigates network latency through speculative video transmission while preventing increased video traffic.

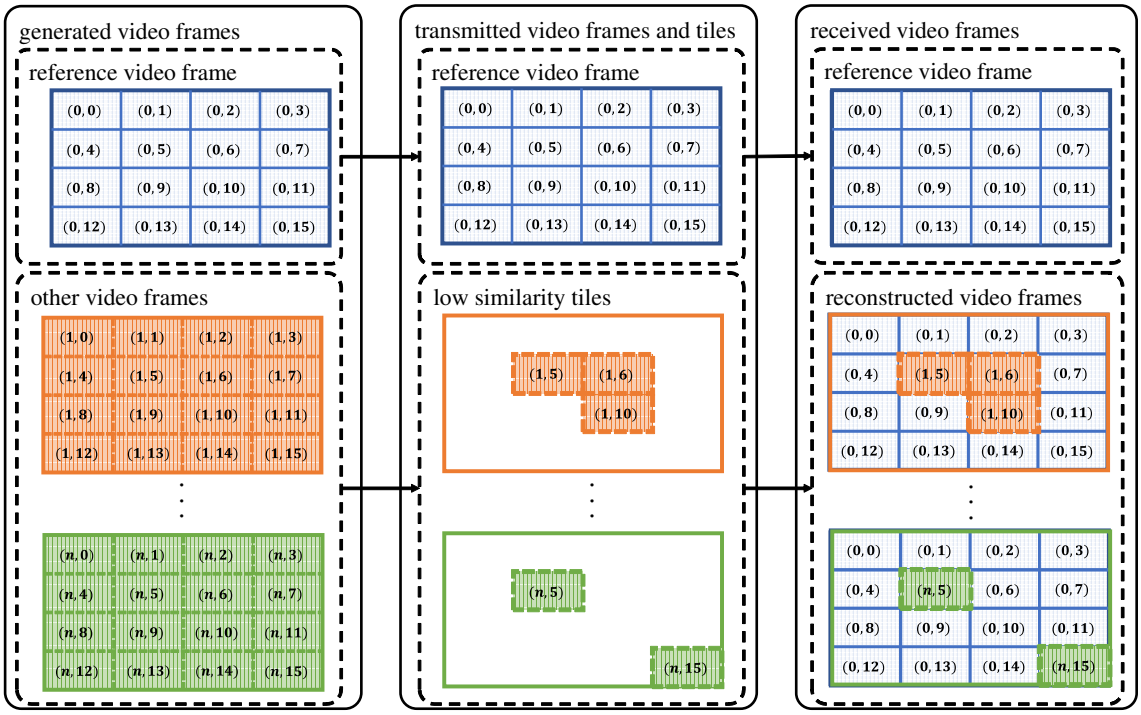


Figure 4.3: Overview of the Proposed Tile-wise Delta Detection

The server detects redundant frames relative to a reference frame and calculates the similarity between the reference video frame and other frames. For this calculation, the video frame is divided into M vertical and N horizontal tiles. The similarity of each tile is computed using the pHash algorithm [95]. The pHash prioritizes extracting low-frequency component features, effectively capturing differences perceivable by humans. It also shows robustness against motion blur and image changes [96], which is why pHash was chosen.

The pHash value calculation process is as follows:

1. Convert the video frame to a grayscale image, maintaining the same luminance as the original color image.
2. Resize this grayscale image to 32×32 pixels.
3. Perform DCT on the resized image and extract low-frequency DCT coefficients over an 8×8 pixel area.
4. Binarize the extracted DCT coefficients based on the median value to generate a 64-bit hash value.

The pHash values of each tile in the reference frame are compared with the hash values of tiles in other video frames. Tiles are identified as similar or different based on the Hamming distance. A more significant Hamming distance indicates a high difference between the data, with the maximum

distance being 64. This method allows for an efficient reduction in the amount of video traffic sent to users.

4.2.3 Number of Speculative Frames Relative to Computational Capability

The speculative video transmission can mitigate the network delay from the user's perspective by sending the video frames of possible user's future input patterns. In the proposed scheme, the cloud gaming server lists the user's potential inputs in future frames and renders the video frames. The cloud gaming server generates the video frame and performs tile-wise delta detection on a frame-by-frame basis. Accordingly, the latency requirement depends on the frame rate of cloud gaming services. We consider the frame rate of 60 fps; thus, the requirement is 16.6 ms.

Let C_S [Hz] be the total operating frequency of the central processing unit (CPU) in the cloud gaming server, P_S [cycles] be the total number of cycles of speculative game execution processes, and f [fps] be the frame rate. Here, A denotes the number of the potential input patterns in each frame. In this case, the possible number of speculative frames $\hat{n}$ [frames] that can be speculatively processed is subject to the following restrictions:

$$\hat{n} = \left\lfloor \log_A \left(\frac{C_S}{f \cdot P_S} \right) \right\rfloor \tag{4.1}$$

Let P_G be the number of game execution process cycles per frame. P_S can be determined as follows:

$$P_S = P_G \cdot A^{\hat{n}} \tag{4.2}$$

Therefore, $\hat{n}$ relative to the total operating frequency of the CPU is as follows:

$$\hat{n} = \left\lfloor \frac{1}{2} \log_A \left(\frac{C_S}{f \cdot P_G} \right) \right\rfloor \tag{4.3}$$

Here, d [s] and t_p [s] denote the network delay and the perceived network delay. When the server lists the potential inputs for future $\hat{n}$ [frames] within each frame, the perceived network delay t_p is as follows:

$$t_p = \begin{cases} 0 & \text{if } \left(\frac{\hat{n}}{f} \geq d \right), \\ d - \frac{\hat{n}}{f} & \text{else} \end{cases} \tag{4.4}$$

The proposed method reduces the perceived network delay by rendering many future input patterns frame-by-frame. However, as defined in Eq. (4.3), significant computation power is required to mitigate the perceived network delay as the value of $\hat{n}$ increases.

4.3 Rate-distortion Optimized Speculative Frame Coding

In this section, we introduce the notion of transition probability $P(F_k)$ to each frame at time t. F_k denotes the k-th frame in the group of frames ($\mathcal{F}_t$) to be speculatively processed at time

t. The frames the user requests are independent and therefore $\sum_k P(F_k) = 1$. The transition probabilities to each frame are assumed to be obtainable in advance. We can determine transition probabilities using time series analysis of user operation logs or machine learning-based analysis. This idea draws influence from the discussion in [81] on improving input prediction based on the assumption that the input continues seamlessly from the immediately preceding input. Note that, depending on the game genre and the input method of the operations, determining the exact transition probabilities might occasionally be practical. However, this limitation does not restrict the generality of our approach, as individuals can modify the probabilistic model without affecting the proposed system.

When the input pattern per frame A is speculatively processed for $\hat{n}$ frames, $\mathcal{F}_t$ is shown as follow:

$$\mathcal{F}_t = \{F_k | k \leq A^{\hat{n}}\} \tag{4.5}$$

Here, we assume that the server has determined the transition probabilities $P(F_k|\mathcal{F}_t)$ for all possible transition frames. Allocating more bits to frames more likely to be displayed by the user improves the user experience.

The rate allocation algorithm is implemented to minimize the distortion expected at the decoder, according to the transition probability $P(F_k|\mathcal{F}_t)$. Assuming $r(k)$ bits are assigned to k-th frames, the distortion is shown as follows:

$$D = \sum_{k=1}^{A^{\hat{n}}} D(r(k))P(F_k|\mathcal{F}) \tag{4.6}$$

In the proposed method, the difference from the reference frame is determined for each tile of M $\times$ N and transmits only those with a difference. τ represents the hash threshold, $r(i,j,k)$ indicates the bitrate of (i,j)-th tile in k-th frame, and $x(i,j,k)$ is a binary variable that determines whether the tile is transmitted or not. Specifically, the proposed scheme assigns $x(i,j,k) = 0$ when the Hamming distance of the tile between the reference video frame and another video frame is below the hash threshold τ and vice-versa. Setting a lower hash threshold increases the number of transmission tiles. In this case, the distortion is defined as:

$$\begin{aligned} D = &\sum_{k=1}^{A^{\hat{n}}} \sum_{i=1}^{\mathrm{M}} \sum_{j=1}^{\mathrm{N}} D(r(i,j,k))P(F_k|\mathcal{F})x(i,j,k) \\ &s.t. \quad \sum_{i=1}^{\mathrm{M}} \sum_{j=1}^{\mathrm{N}} \sum_{k=1}^{A^{\hat{n}}} r(i,j,k)x(i,j,k) \leq R_{limit} \\ &s.t. \quad x(i,j,k) = \begin{cases} 1 & \text{(if send)} \\ 0 & \text{(else)} \end{cases} \end{aligned} \tag{4.7}$$

Here, $r(i,j,k)x(i,j,k)$ denotes the rate distribution limited by the given rate R_{limit}, $D(r(i,j,k))$ denotes the distortion for each tile encoded with $\mathbf{r}(i,j,k)$ bits. Since $P(F_k|\mathcal{F})$ does not depend on

Table 4.1: Relationship between SSIM and MOS [1]

MOS	SSIM
5 (excellent)	≥ 0.98
4 (good)	< 0.98 and ≥ 0.95
3 (fair)	< 0.95 and ≥ 0.88
2 (poor)	< 0.95 and ≥ 0.88
1 (bad)	< 0.88

Table 4.2: The SSIM index and traffic reduction ratio in each commercial game.
The proposed method divided each video frame into 3×3 tiles and set τ=0 for tile-wise delta detection.

Types	first-person game			third-person game			omniscient game		
Titles	Valorant	OW2	Borderlands3	Genshin Impact	MHW	ELDENRING	SFV	Hearthstone	Cuphead
SSIM	0.9848	0.9815	0.9609	0.9802	0.9682	0.9533	0.9755	0.9863	0.9782
Reduction ratio [%]	35.29	4.17	23.18	17.24	7.99	24.97	52.10	54.14	44.00

(a) Original (Valorant)

(b) Proposed (Valorant)
SSIM index is 0.9848
Reduction ratio is 35.29 %

Figure 4.4: Valorant snapshots comparing original and proposed methods. The proposed method's SSIM index and reduction ratio are shown, highlighting the efficiency and quality preservation.

r, the rate distribution optimisation problem is solved with Lagrange multipliers $\lambda > 0$, the cost J is as follow:

$$J = \min_{\mathbf{r},\mathbf{x}} \left\{ \sum_{k=1}^{A^{\hat{n}}} \sum_{i=1}^{\mathrm{M}} \sum_{j=1}^{\mathrm{N}} D(r(i,j,k)) P(F_k|\mathcal{F}) x(i,j,k) + \lambda ||\mathbf{rx}||_1 \right\} \tag{4.8}$$

By solving the optimization problem, the proposed scheme can find the optimal hash threshold τ and bit assignment for each tile $r(i,j,k)$ to minimize the distortion under the given rate R_{limit}.

4.4 Evaluation

4.4.1 Traffic Reduction in Commercial Games

We measured the video traffic and the corresponding structural similarity (SSIM) index [1] of the speculative video transmission with tile-wise delta detection. This evaluation assumes that the delay is zero due to speculative execution, so the only factor that affects the user's perceived quality is the video quality. SSIM index is a better objective metric for predicting the perceptual

(a) Original (Genshin Impact)

(b) Proposed (Genshin Impact)
SSIM index is 0.9767
Reduction ratio is 23.77 %

Figure 4.5: Genshin Impact snapshots showing the effectiveness of the proposed method. The SSIM index and reduction ratio indicate significant improvements in performance and quality.

(a) Original (SFV)

(b) Proposed (SFV)
SSIM index is 0.9741
Reduction ratio is 52.95 %

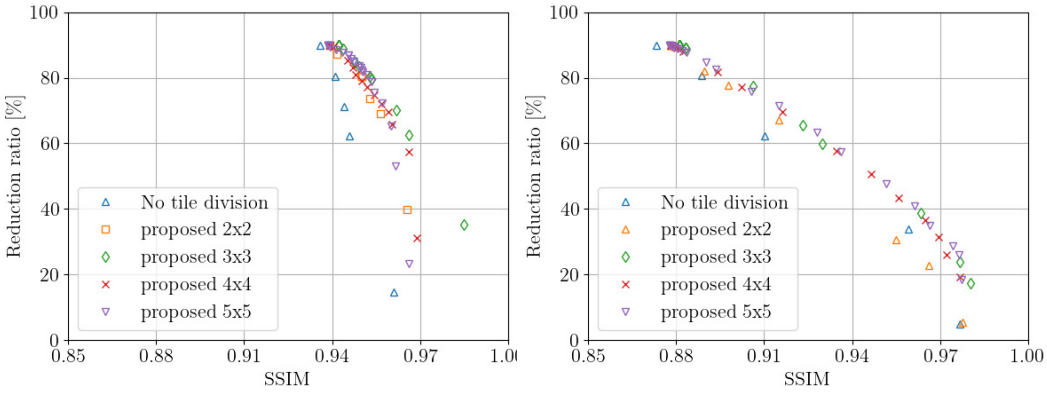
Figure 4.6: SFV snapshots comparing the original and proposed techniques, with a focus on the SSIM index and reduction ratio, demonstrate the effectiveness of the proposed method.

similarity between original and processed video frames. Table 4.1 presents the relationship between SSIM and MOS. Larger values of the SSIM index close to 1 indicate higher perceptual similarity between these frames.

This evaluation classifies commercial games into first-person, third-person, and omniscient. First-person games are displayed from the viewpoint of the user-controlled character. FPS is a typical first-person game. The video frame of an FPS game is characterized by the display of the user interface and weapons, such as guns and knives held by the characters at specific coordinates. The objects contained within the viewpoint change rapidly in response to the user's inputs.

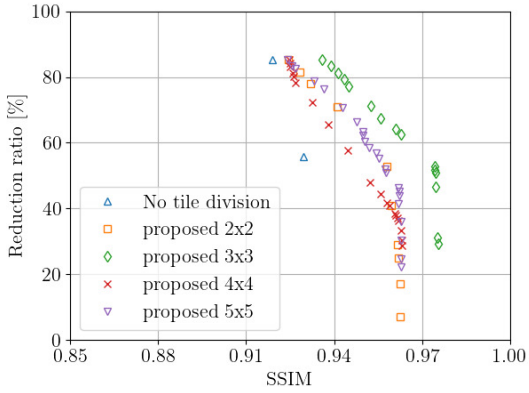
Third-person games are displayed from the rear view of the user's character. A typical third-person game is an action game. The video frame of an action game has the characteristic of displaying the user's character at specific coordinates. As in FPS games, the objects contained within the viewpoint change rapidly in response to the user's inputs.

Omniscient games are displayed so that the user overlooks the controlling character. A typical



(a) first-person game (Valorant)

(b) third-person game (Genshin Impact)



(c) omniscient game (SFV)

Figure 4.7: Traffics relative to SSIM index. To evaluate traffic based on the SSIM index, we measured the traffic and average SSIM index for each tiling by varying the Hamming distance threshold τ value from 0 to 64 for the original video frame.

omniscient game is a fighting game. The video frame of a fighting game displays a fixed viewpoint such that the user's character and the entire field of play are contained. The user-controlled character moves within a fixed viewpoint in response to the user's inputs.

We used the following commercial games for comparison: Valorant [97], Overwatch2 (OW2) [98], and Borderlands3 [99] as first-person games; Genshin Impact [100], Monster Hunter World (MHW) [101], and ELDENRING [102] as third-person games; and Street Fighter V (SFV) [103], Hearthstone [104], and Cuphead [105] as omniscient games. To facilitate comparison, we replicated a game state for each game and obtained video frames corresponding to various inputs from that state. Specifically, we recorded 10 video frame patterns for each game, from which one frame was chosen randomly as the reference frame. All frames, also divided tiles, were processed without compression. Accordingly, in Equation (4.8), $P(F_k|\mathcal{F})$ remains constant regardless of the values of k, and $D(\mathbf{r}(i,j,k))$ equals zero.

Table 4.2 shows each commercial game's SSIM index and traffic reduction ratio on the proposed system. In this evaluation, the proposed method divided each video frame into 3×3 tiles, and the Hamming distance threshold is $\tau = 0$, i.e., only tiles with perfect matches are not transmitted. The SSIM index was measured with reference to the original video frame. As a result, all omniscient games had a reduction efficiency of 40–50 % with an SSIM index of approximately 0.98. Since the region in which the user input effect tends to be limited, the tiles with zero Hamming distances are more likely to appear. Therefore, high reduction efficiency can be achieved. On the other hand, some first-person and third-person games had video quality degradation and low reduction ratio compared to omniscient games. In first-person and third-person games, the entire video frame may change owing to user input. Therefore, the reduction efficiency is reduced when the Hamming distance threshold is set to $\tau = 0$. Optimizing the Hamming distance threshold and the number of tile divisions may improve reduction efficiency while limiting quality degradation.

Figures 4.4–4.6 show the snapshots of the games. We select Valorant, Genshin Impact, and SFV as the first-person, third-person, and omniscient games. Figures 4.4(a), 4.5(a) and 4.6(a) show the original video frames and Figures 4.4(b), 4.5(b) and 4.6(b) show the proposed video frame. Here, we divided each uncompressed frame into 3×3 tiles and set τ=0 for the first-person, third-person, and omniscient games, respectively. From the snapshots, the proposed method can reduce video traffic irrespective of the game types. In addition, quality degradation due to the tile-wise delta detection does not significantly impact the visual quality.

Finally, Figures 4.7(a) through 4.7(c) show the reduction ratio for the SSIM index of the first-person, third-person, and omniscient games, respectively. In this evaluation, the proposed method divided each video frame into 2×2 to 5×5 tile divisions and varied the Hamming distance threshold τ from 0 to 64. As the value of τ increases, the number of tiles regarded as similar also increases, leading to higher video quality and larger SSIM indexes. Here, no tile division means that the similarity calculation is performed on the whole frame without tiling it.

We can see the following results:

- Raising the threshold τ value enhances the traffic reduction ratio across all games by classifying more tiles as similar. Nevertheless, this also leads to a significant decrease in the SSIM index.

- Figures 4.7(a) to (c) show that performing similarity estimation on each tile after division performs better compared to the scenario where similarity estimation is performed without division. All games produced the highest quality video, especially when split into 3×3 tiles.

- Fig. 4.7(c) indicates that optimizing the number of tiles increases the reduction ratios. For instance, with an SSIM index of 0.9628, the reduction ratio registers 62.55 % for 3×3 tile division and 36.14 % for 4×4 tile division.

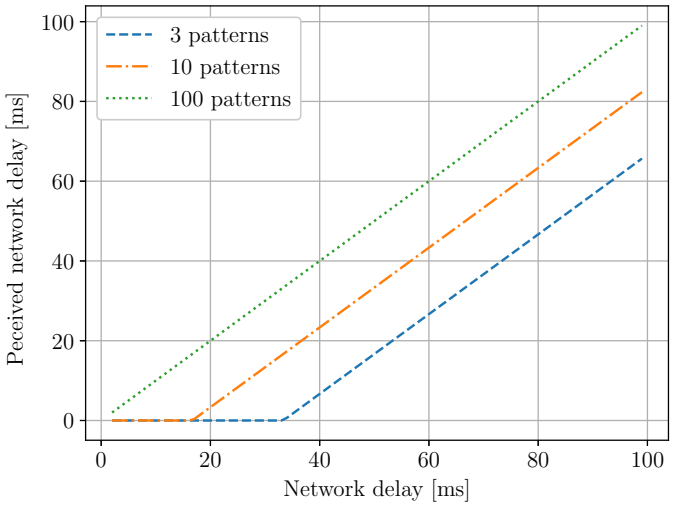


Figure 4.8: The perceived network delay as a function of network delays with the different number of potential input patterns in each frame.

- Fig. 4.7(c) shows that tile-wise delta detection works well in omniscient games. In such games, the background tiles are static, irrespective of the user's inputs, which can lead to a large reduction in traffic.

- Fig. 4.7(a) indicates the potential effectiveness in other genres, depending on each game's characteristics. In other words, the proposed method works well when the game field has minimal transitions for each input, such as when the background is monotonous.

Considering the rate-distortion optimization discussed in Section 4.3, the results from Figures 4.7(a) to 4.7(c) suggest that regardless of the number of tile divisions, setting a smaller τ increases the number of transmitted tiles, i.e., there is a higher proportion of instances where $x = 1$. In such cases, it becomes necessary to reduce the quality of each tile through rate-distortion optimization, thereby decreasing the value of r. Conversely, setting larger τ decreases the number of transmitted tiles, i.e., there is a higher proportion of instances where $x = 0$. The value of r can be increased through rate-distortion optimization.

4.4.2 Feasibility

In this chapter, we conduct experiments to discuss the perceived network delay under the different computation resources and the number of potential input patterns in each frame. Table 3.1 shows the experimental setup. Intel Core i9-10850K has ten unlocked cores and hyper-threading, and each core can turbo up to an operating frequency of 5.2×10^9 Hz. Here, we consider the average number of game execution process cycles per frame P_G to be 2.5×10^6 cycles based on the measurement during running SFV.

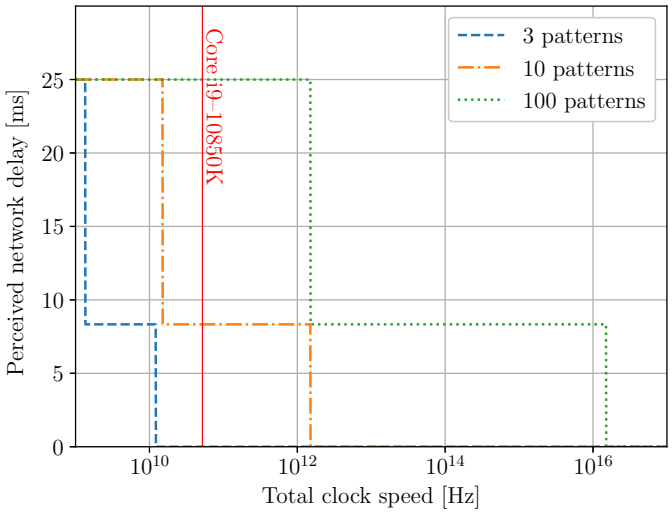


Figure 4.9: The perceived network delay as a function of the total clock speed, where the network delay is assumed to be 25 ms.

Fig. 4.8 shows the perceived network delay as a function of network delays with the different number of potential input patterns in each frame. We assume the same computation capability as the Intel Core i9-10850K CPU used in our implementation. From the evaluation results, the cloud gaming server can decrease the number of video frames needed to be rendered in a short network delay environment and a limited number of potential inputs. In this case, the cloud gaming server can complete the required operations in every frame, and the perceived network delay becomes zero. In a long network delay and many of the user's potential inputs, the cloud gaming server does not complete the required operations within one frame, and thus, the perceived network delay becomes longer. However, the proposed scheme can reduce the perceived network delay because the cloud gaming server sends the rendered video frames once the required operations are finished. A lower perceived network delay, i.e., RTT, improves the cloud gaming experience.

Fig. 4.9 shows the perceived network delay as a function of the total clock speed, where the network delay is assumed to be 25 ms [56]. When the number of potential input patterns is three, the proposed scheme can eliminate the network delay using an off-the-shelf CPU, e.g., Intel Core i9-10850K. On the other hand, the proposed scheme needs 100 and 10000 times the computational capabilities of the current experimental setup to achieve zero perceived network delay when the number of potential input patterns in each frame is 10 and 100, respectively.

4.5 Conclusion

This chapter proposes a traffic reduction method for speculative video transmission in cloud gaming systems to mitigate network delay. The concept of the proposed method can be applied to all

genres of commercial games. Evaluations on Valorant, Genshin Impact, and SFV showed that the proposed method reduced the traffic by approximately 35 %, 24 %, and 53 %, respectively, without video compression techniques when the average SSIM index was approximately 0.98. In the future, we will extend the proposed scheme to accommodate multiple users. In this case, the cloud gaming server lists future inputs for each user and renders the video frames for each user. A key issue is reducing the traffic increment with an increase in the number of users. A potential solution is to employ a tile-wise delta detection for the video frames across the users to remove redundant information. How to efficiently remove the redundant information to accommodate multiple users is left as the future work.

In addition, we discussed the feasibility of the proposed method based on the experimental environment. The proposed speculative execution for two frames may be possible with three input patterns in each frame. Current cloud gaming systems generally operate at 60 fps. But it will be necessary to consider higher fps, such as 200 or 300 fps, in the future. Ideal speculative execution requires processing all possible patterns during one frame. In high frame rate environments such as fighting games and FPS games, which require real-time performance, game processing is usually performed on a frame-by-frame basis. As the time between one frame becomes shorter, the limit on the number of patterns that can be processed becomes more severe. Even if we aim to reduce the response delay by performing speculative execution to the extent possible, the effect of the proposed method is likely to be smaller at high frame rates because the time to be reduced will be limited. On the other hand, it is possible to consider a game that complements frames to improve visual smoothness. It may be effective to use an approach that simultaneously improves video quality by interpolating video to higher fps, such as 200 or 300 fps while keeping the speculative execution frequency at 60 fps using the proposed method.

Chapter 5

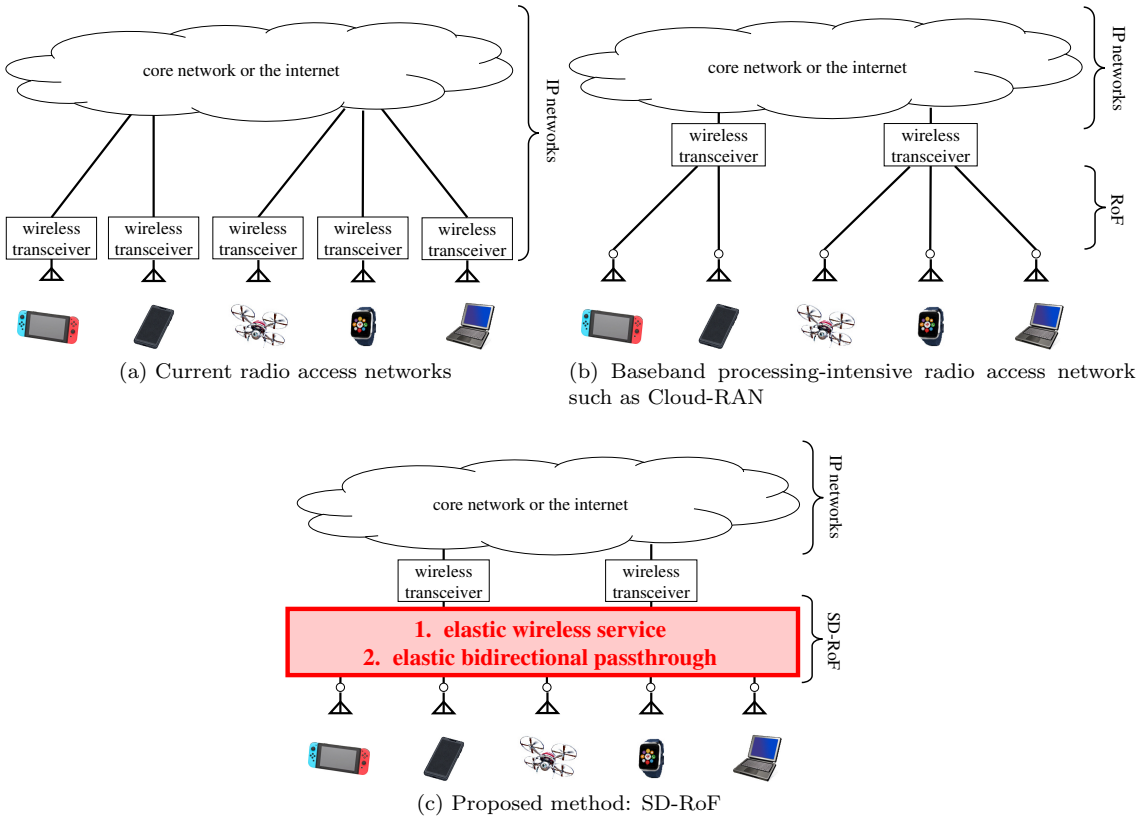
Low-latency Cloud Gaming System Architecture

5.1 Introduction

In Chapter 5, we explore the deployment of cloud gaming systems on mobile networks, tackling unique challenges inherent in this environment. Our previous chapters have concentrated on using speculative execution to reduce response latency from an application perspective in cloud gaming systems. However, applying speculative cloud gaming to mobile networks introduces distinct challenges, primarily due to the higher latency often associated with these networks compared to wired alternatives.

Addressing the inherent response latency from a network architecture perspective is crucial for successfully deploying speculative cloud gaming systems on mobile networks. Reducing response latency minimizes the delay that must be concealed from the user, enhancing the potential for achieving near-zero latency and boosting the system's overall efficiency. Established methods for reducing response latency in cloud gaming systems, like edge computing and adaptive bitrate control, face adaptation challenges. These challenges are rooted in computational costs, as detailed in Chapter 3, and the limitations of current video compression techniques, as discussed in Chapter 4. Therefore, a fundamental revision of the network architecture is essential.

We propose the software-defined radio over fiber (SD-RoF) architecture in response to these challenges. SD-RoF seamlessly integrates optical and wireless technologies to provide a Radio Access Network (RAN) with large capacity, low latency, and no protocol restrictions. We illustrate the differences between existing wireless access networks and the proposed SD-RoF in Figure 5.1. Figure 5.1(a) depicts typical existing wireless access networks established using IPs with wireless transceivers operating as base stations at the network edges. Recent research, such as that on cloud radio access networks (C-RANs) shown in Figure 5.1(b), has focused on centralized wireless access networks for more flexible communication. Examples include FlexRAN [106], SoftAir [107],



(a) Current radio access networks

(b) Baseband processing-intensive radio access network such as Cloud-RAN

(c) Proposed method: SD-RoF

Figure 5.1: Comparison of existing and proposed systems

H-RAN [108], and F-RAN [109].

The principles of SDN inspire SD-RoF. The SDN architecture distinguishes between a control plane and a data plane, centralizing control of network devices via a controller. This structure enables flexible network configuration and operation, a concept explored in [110]. SDN applications range from network slicing [111] to virtual private networks [112], and even hybrid setups with traditional IP networks [113]. SD-RoF extends these principles to wireless access networks, offering detailed software control over the interactions between antennas and wireless transceivers. Figure 5.1(c) demonstrates this control by detailing the RoF component's integration in the baseband processing of centralized wireless networks.

The primary distinction between SD-RoF and conventional SDN is their focus on different network types and layers of operation. SDN targets wired networks and typically handles network configuration and switching at the data link layer or higher. In an SDN framework, switches direct packets based on rules set by the controller. Conversely, SD-RoF targets wireless access networks and establishes and switches them at the physical layer. Conversely, SD-RoF utilizes controllers to modify network configurations by dynamically altering paths in optical switches.

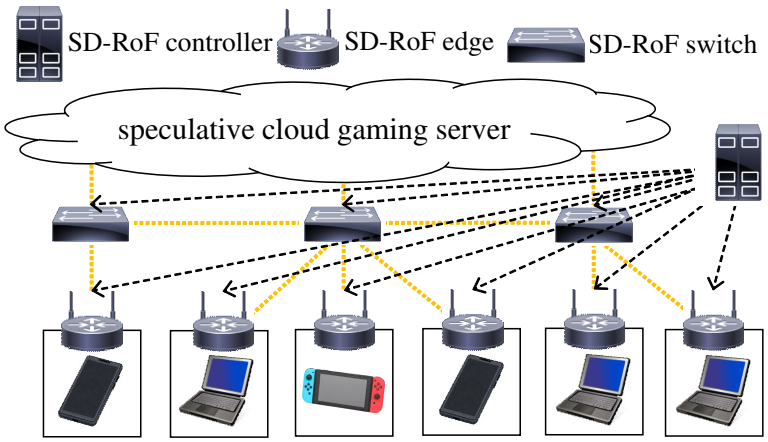


Figure 5.2: SD-RoF architecture

SD-RoF supplants the current IP-based wireless access network with a software-defined RoF network, combining optical and electrical switches. It introduces two key functions: elastic wireless service and elastic bidirectional passthrough. The elastic wireless service enables users to access required wireless services anytime and anywhere. The elastic bidirectional passthrough connects two remotely located points via RoF using radio waves, facilitating bidirectional radio communication between devices at each location.

5.2 SD-RoF: Software-Defined Radio over Fiber

5.2.1 Architecture Overview

In cloud gaming systems, it is essential to anticipate various wireless communication standards for client-side devices. Network equipment traditionally adjusts to these standards, often leading to processing delays. We advocate for direct end-to-end communication to significantly reduce response times.

Our study identifies four critical requirements for the network architecture of cloud gaming systems:

1. **Minimizing Propagation Delay:** Utilizing an optimal path optical network to ensure the fastest signal travel.

2. **Reducing Transmission Delay for High-Quality Video:** Establishing a high-capacity communication path for quick and clear video transmission.

3. **Eliminating Processing and Congestion Delays:** Ensuring seamless data flow without any processing hiccups on the communication path.

4. **Facilitating Bidirectional P2P Communication:** Enabling the conversion of offline LAN games to online formats, enhancing the gaming experience.

In a low-latency network architecture, effectively addressing propagation and transmission delays is critical. The system must make optimal routing decisions and provide high-bandwidth communication channels. Additionally, transmitting wireless communication signals without any processing and congestion delays is a priority. Moreover, we see significant potential in enabling bidirectional P2P communication, given cloud gaming's evolving requirements. Current cloud gaming systems are mostly single-player. Therefore, we must devise a method to send controller signals directly for local multiplayer or offline games in the cloud. We can broaden our applications by enabling cloud gaming systems to convert offline LAN games to online formats.

We propose SD-RoF, a network that integrates optical and wireless technologies seamlessly. Figure 5.2 illustrates the SD-RoF architecture, notable for its diverse requirements, high bandwidth capacity, and low latency. SD-RoF utilizes analog RoF in its communication path, leveraging RoF's broadband capabilities for direct analog transmission of wireless signals, effectively reducing processing delays. By significantly lowering the response delays users typically face, we can streamline speculative execution, reducing computational costs.

In SD-RoF, each wireless spectrum space is defined as an area for direct device communication. Typically, devices in different wireless spectrum spaces cannot communicate directly. SD-RoF overcomes this limitation, allowing direct communication between these spaces, enabling bidirectional P2P communication, and expanding cloud gaming systems' applications. The network includes SD-RoF switches and edges, an SD-RoF controller, wireless transceivers, and RoF, directly connecting specific wireless spectrum spaces.

This chapter will detail each component required for SD-RoF, explaining their roles and functionalities in creating an effective low-latency cloud gaming system.

5.2.2 RoF

RoF connects two electro-optic (E/O) converters via optical fiber, which is essential in modern telecommunication. It modulates the intensity of optical signals with radio signals and transmits them through the fiber. Renowned for its high bandwidth and low latency, RoF is a staple in transmitting large-capacity data from mobile network cores to edges. In 6G networks, RoF becomes even more crucial for transmitting high-attenuation signals like mmWave and terahertz communications [114].

RoF has two variants: Digital RoF (D-RoF) and Analog RoF (A-RoF). D-RoF, transmitting digital signals through light on-off keying, resists nonlinear distortion and noise from optical transmitters. However, its bidirectional communication requires additional Analog-to-Digital/Digital-to-Analog (AD/DA) converters and frequency converters, which add latency and cost. On the

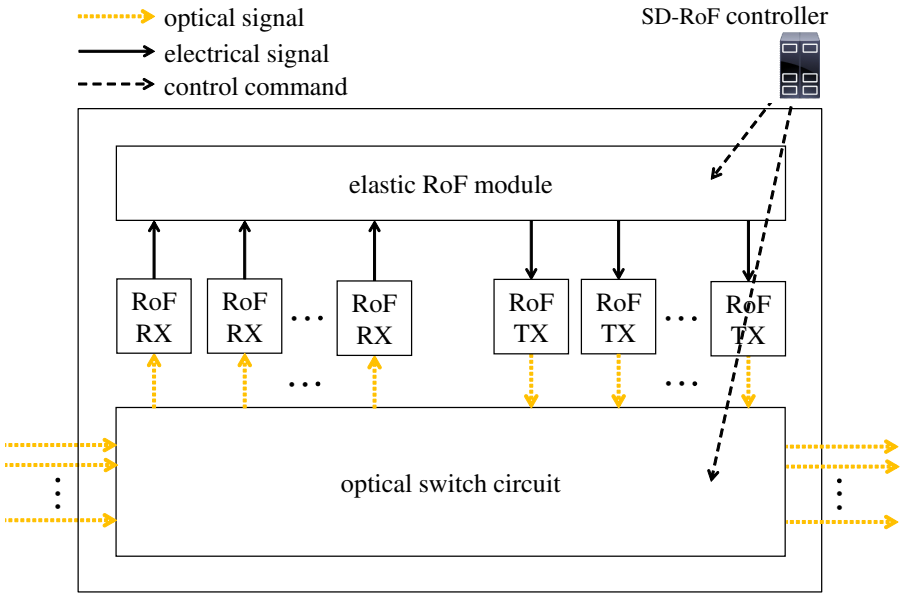


Figure 5.3: SD-RoF Switch

other hand, A-RoF directly converts analog electrical signals into optical signals. This process allows for low-latency and wide-bandwidth communication, making A-RoF versatile across different wireless standards within its frequency range. The A-RoF transceivers employed in our SD-RoF prototype offer a wide bandwidth from 40 MHz to 6 GHz and are cost-effective, priced at only a few hundred dollars.

Given these factors, we chose Analog RoF for the SD-RoF architecture. A-RoF's low latency, cost-effectiveness, and wide communication bandwidth align perfectly with our needs. Additionally, its ability to support various wireless standards without changing equipment makes it an ideal choice for a flexible and efficient cloud gaming system.

5.2.3 SD-RoF Switch

Figure 5.3 shows the configuration of the SD-RoF switch. The SD-RoF switch comprises an optical switch circuit, an Elastic RoF module, and RoF transceivers. The optical switch circuit switches the input and output of optical signals based on control commands from the SD-RoF controller. This switch changes the one-to-one relationship between input and output ports without involving packet processing. Hence, the same configuration can be used regardless of the input signal format or data rate.

The Elastic RoF module is an essential component of the SD-RoF switch, and its detailed functions will be introduced in Chapter 5.2.5. RoF transceivers perform the mutual conversion of optical and electrical signals in the SD-RoF switch. The RoF receiver converts wireless com-

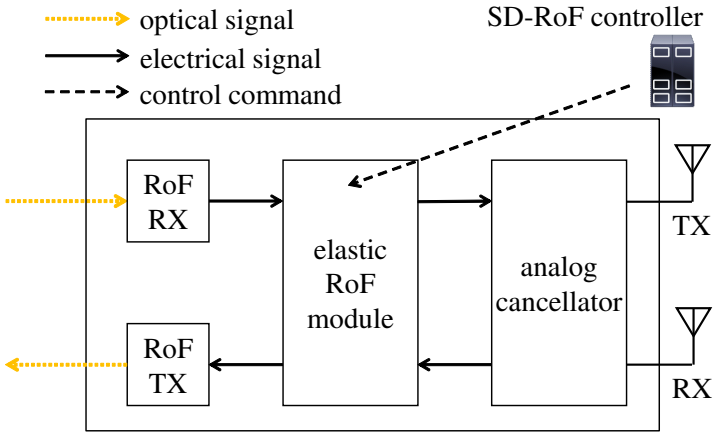


Figure 5.4: SD-RoF Edge

munication signals into electrical signals, which are frequency-converted and multiplexed by the Elastic RoF module. Finally, the RoF transmitter converts these signals back into optical signals for transmission. This process bundles wireless communication signals from multiple paths into a single optical signal, enabling efficient use of optical frequency resources.

5.2.4 SD-RoF Edge

Figure 5.4 shows the configuration of the SD-RoF Edge. The SD-RoF Edge comprises a RoF transceiver, an Elastic RoF module, an analog canceller, and a transceiver antenna. The Elastic RoF module, similar to the one used in the SD-RoF switch, is essential for handling various communication standards in the SD-RoF Edge.

The Elastic RoF module provides a vital function of efficiently managing signals in different frequency bands. For instance, MIMO transmission can process multiple signals simultaneously by converting them into separate frequencies and transmitting them as a single optical signal to another SD-RoF Edge. This process is crucial for preventing interference and collisions of wireless signals.

The role of the analog canceller is to work in conjunction with the Elastic RoF module to reduce signal interference and improve signal quality. The specific mechanism and applications are explained in detail in Chapter 5.2.6. By utilizing communication through the SD-RoF Edge, it is possible to process signals in both transmission and reception directions, enhancing the system's flexibility and efficiency.

5.2.5 Elastic RoF Module

The Elastic RoF module, shown in Figure 5.5, performs frequency conversion and multiplexing of signals in SD-RoF Edges and SD-RoF Switches. This module comprises a control command

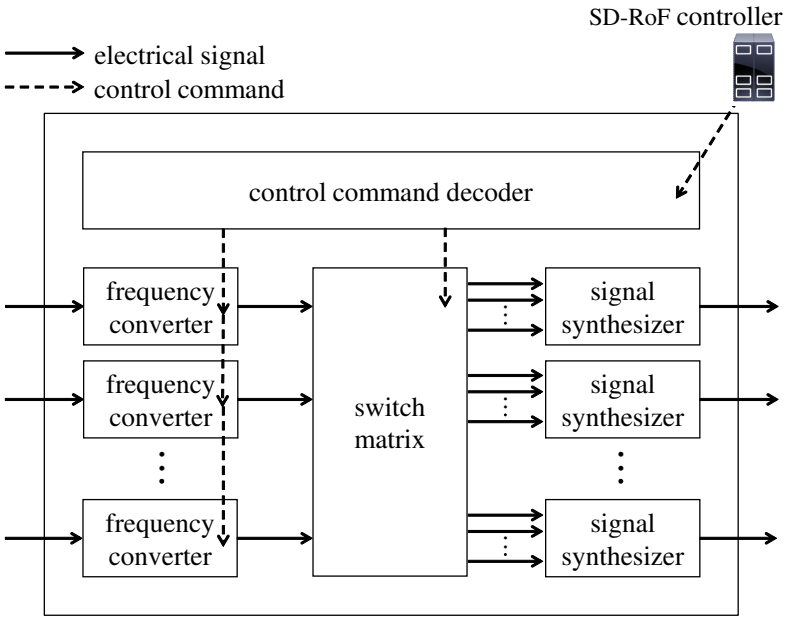


Figure 5.5: Elastic RoF Module

decoder, frequency converters, switch matrices, and signal synthesizers. The control command decoder receives instructions from the SD-RoF controller and appropriately sets the frequency converters and switch matrices.

Figure 5.6 illustrates the structure of a frequency converter. The frequency converter changes the carrier frequency of the input signals. It avoids collisions with other communications by converting the signal frequency. The input signal is initially quadrature demodulated at carrier frequency f_1 and then converted into a baseband signal using a Low Pass Filter (LPF). This baseband signal is then modulated at carrier frequency f_2 for frequency conversion.

Regarding the signal flow, the Elastic SD-RoF Module initially distributes the input signal to either the frequency converter or the switch matrix. If frequency conversion is required, the signal passes through the frequency converter and then goes to the signal synthesizer via the switch matrix. If frequency conversion is not needed, the signal directly enters the switch matrix and, from there, goes to the signal synthesizer. The signal synthesizer overlays multiple input signals to produce a single output signal. For example, if signals $s_1(f_1), s_2(f_2), \cdots, s_n(f_n)$ are input, it outputs the signal $s_1(f_1) + s_2(f_2) + \cdots + s_n(f_n)$. Each signal $s_k(f_k)$ at center frequency f_k is defined so that it does not overlap with any other k.

5.2.6 Analog Cancellator

In our proposed network architecture, two SD-RoF edges are connected by two RoFs, an upstream and a downstream, expanding the wireless communication area. Devices connected to the SD-RoF

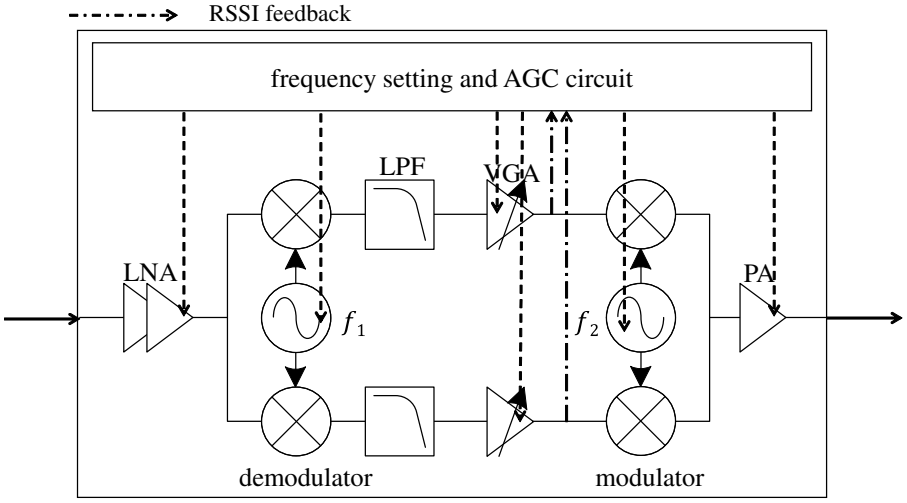


Figure 5.6: Frequency Converter

edges can communicate as if they are in the same wireless communication area. However, simply connecting two antennas does not enable bidirectional communication. Figure 5.7 illustrates the signal loop problem that occurs when extending the wireless communication area by connecting two SD-RoF edges. The signal from terminal α is received by the receiver antenna of SD-RoF edge α and transmitted from the transmitter antenna of SD-RoF edge β via the RoF. Conversely, the signal from terminal β is received by the receiver antenna of SD-RoF edge β and transmitted from the transmitter antenna of SD-RoF edge α via the RoF. When the signal from terminal α is transmitted by the transmitter antenna of SD-RoF edge β, the receiver antenna of SD-RoF edge β receives it as a self-interference signal. If the self-interference signal is strong, it can create an infinite loop, hindering communication.

Our proposed method combines a calibration circuit and an analog cancellator to resolve the signal loop issue. Figure 5.8 shows an overview of the analog cancellation of the radio signal loop between two SD-RoF edges. The loop signal illustrated in Figure 5.7 is canceled by equipping the SD-RoF edge with the analog cancellator shown in Figure 5.9. The analog cancellator we use is a passive cancellation circuit developed by Kobayashi et al. [4] for full-duplex wireless communication.

Figure 5.9 details the structure of the analog cancellator. The cancellator generates a signal with the same amplitude and antiphase as the input signal, canceling the loop signal by overlaying it with the signal received by the receiving antenna. The calibration circuit shown in Figure 5.9 adjusts the passive elements of the cancellator. A transmitter in the calibration circuit sends out an unmodulated continuous wave, reproducing the multipath of the radio transmission path between the transmitter and receiver antennas. The parameters of each passive element are set passively by the micro-controller unit (MCU). The MCU receives the value of the power sensor included in

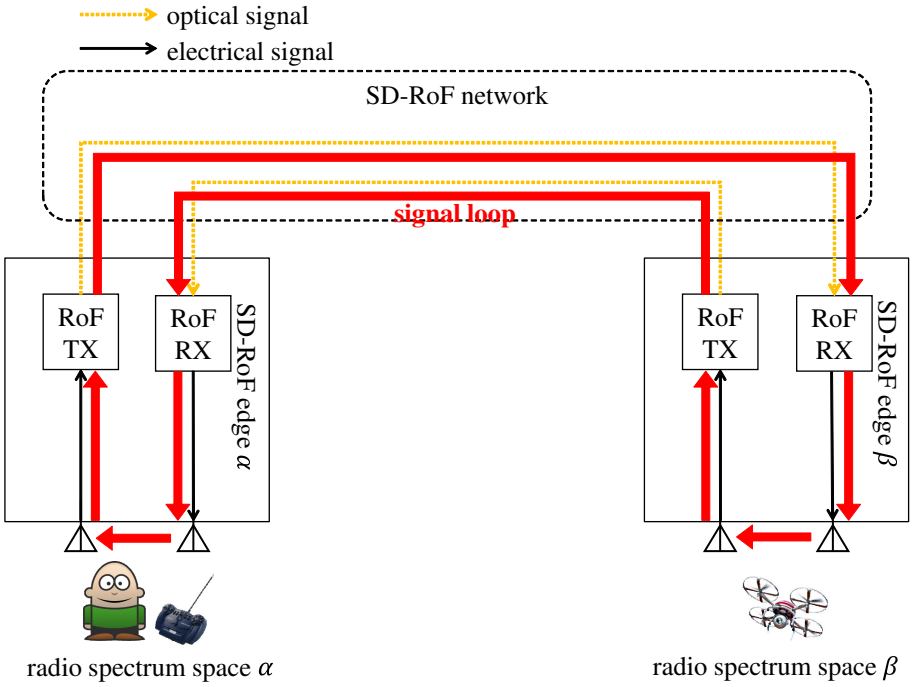


Figure 5.7: Radio signal loop between two SD-RoF edges

the calibration circuit and sets the parameters to maximize the cancellation amount. Therefore, periodic calibration may be necessary in environments with significant changes, such as outdoor settings. The time required for each calibration process is approximately 3.3 milliseconds.

5.3 Evaluation

5.3.1 Performance Analysis

Scalability

To quantify the scalability of the SD-RoF network, we evaluated the impact of implementing Elastic RoF modules by examining the number of fiber optic cables between the networks. The SD-RoF network model shown in Figure 5.10 assumes that all radio spectrum spaces communicate over a 2.412 GHz wireless LAN. We compared the number of fiber optic cables required for interconnections between SD-RoF switches.

We assessed the performance of SD-RoF switches under three scenarios: with Elastic RoF modules, with Wavelength Selective Switches (WSS) but without Elastic RoF modules, and either. WSS, capable of multiplexing and demultiplexing optical signals in the optical domain, separates WDM signals for each wavelength and directs them to chosen output ports. Figure 5.11 displays the comparison results. Implementing Elastic RoF modules demonstrated significant improvement in scalability, maintaining a constant number of fiber optic cables for up to 32 interconnections.

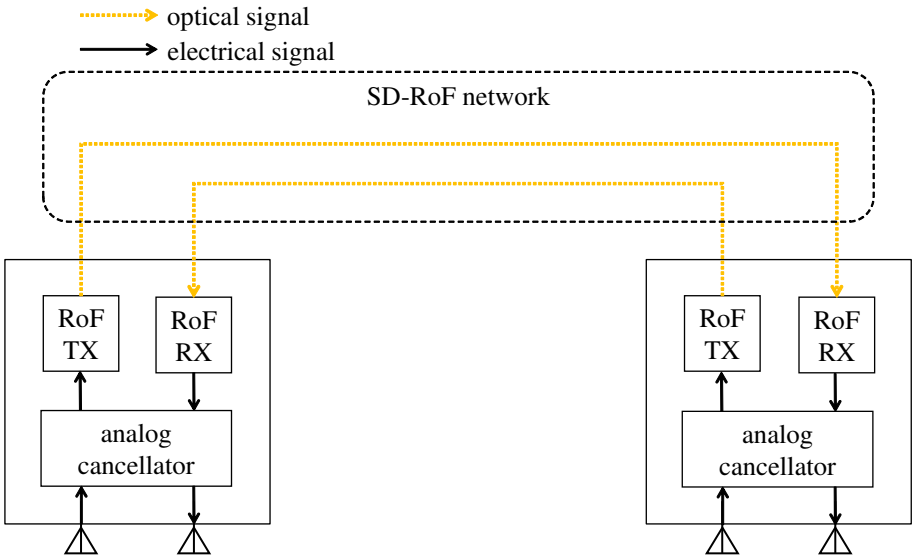


Figure 5.8: Analog cancellation of radio signal loop between two SD-RoF edges

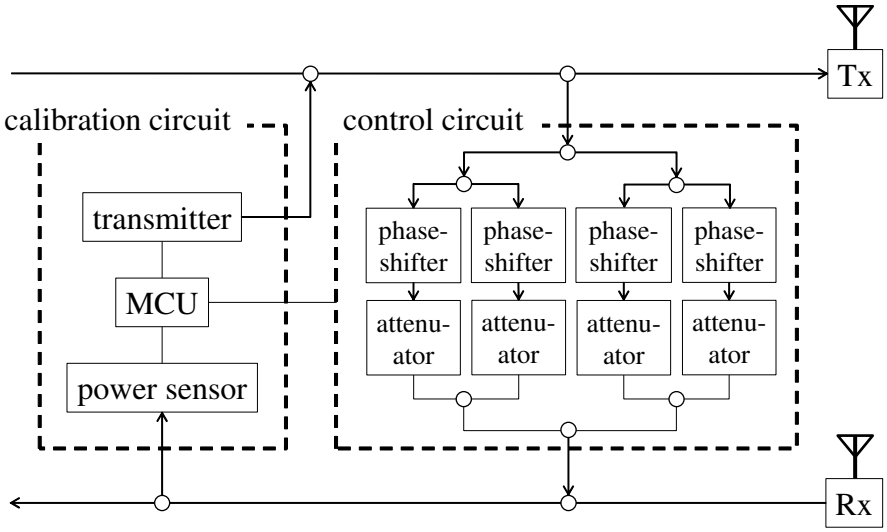


Figure 5.9: Configuration of the analog cancellator

This finding highlights the substantially enhanced flexibility and scalability of the SD-RoF network with Elastic RoF modules. Enhancing flexibility and scalability is crucial for the design of SD-RoF networks to meet future large-scale communication demands. Within the bandwidth limitations supported by the RoF transceivers, the Elastic RoF module efficiently accommodates up to 100 interconnections through a single fiber optic cable, streamlining the construction and operation of large-scale SD-RoF networks.

From this evaluation, the design of SD-RoF networks incorporating Elastic RoF modules plays

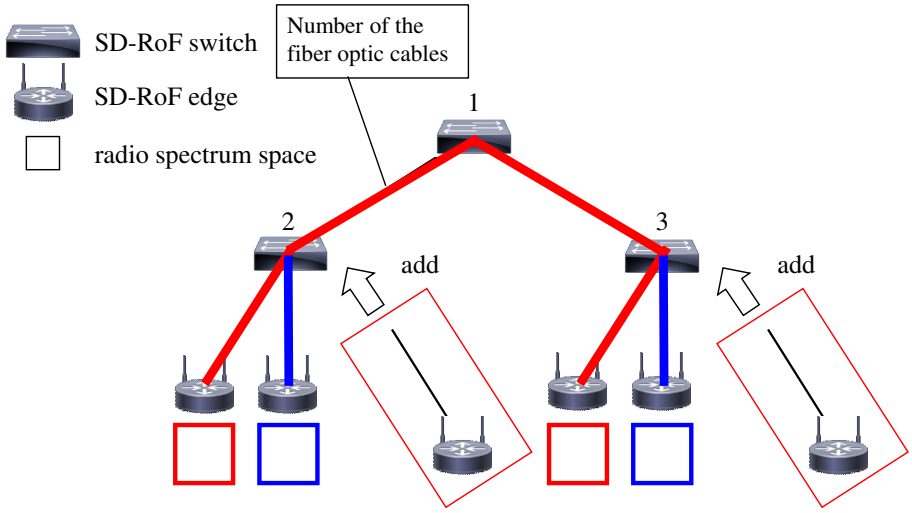


Figure 5.10: SD-RoF network evaluation model

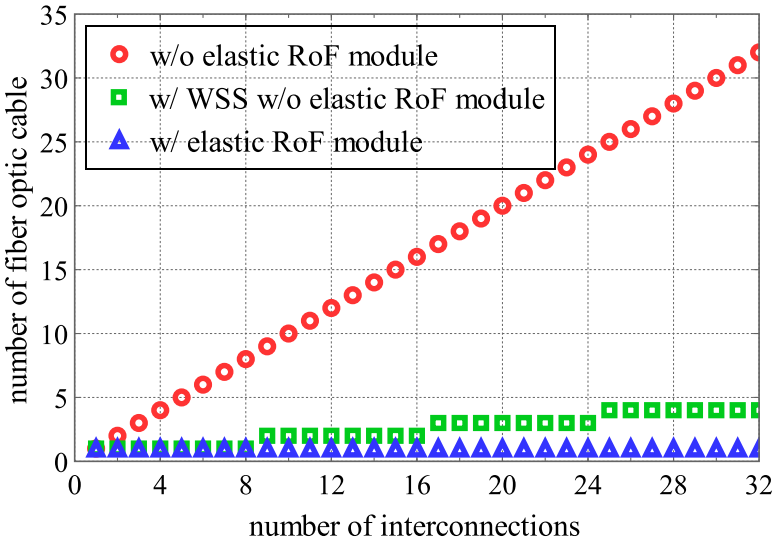


Figure 5.11: Number of fiber optic cables relative to the number of interconnections

a vital role in the future development and expansion of wireless communication networks. This approach allows for expanding wireless communication areas while minimizing the costs and complexity of communication infrastructure, a key consideration in deploying SD-RoF networks.

Signal-to-Noise Ratio (SNR)

We evaluated the SNR of elastic bidirectional passthrough to describe the effect of deploying an analog cancellator. We considered P_{tx} [dBm], which denotes the power of the signal transmitted from the terminal in base α, $L_{\mathrm{rf}}(d)$ [dB] denotes the attenuation when the distance in the wireless communication is d [m], A_{rof} [dB] denotes the signal amplification rate of the RoF transmission module (RoF Tx), $\mathrm{nF}_{\mathrm{rof}}$ [dB] denotes the noise figure in the RoF, $L_{\mathrm{opt}}(d)$ [dB] denotes the attenua-

tion for optical communication over a distance d [m], L_{ant} [dB] denotes the attenuation between the transmission and reception antennas in the SD-RoF edge, C [dB] denotes the cancellation amount of the self-interference cancellation circuit, S_n [dBm] denotes the signal's power that reaches the transmission antenna point in the n-times loop, and N [dBm] denotes the noise floor. S_0 is the strength of the initial signal that reaches the transmission antenna point in the SD-RoF edge β from the SD-RoF edge α. Subsequently, S_0 is determined as follows:

$$S_0 = P_{\text{tx}} - L_{\text{rf}}(d_\alpha) + A_{\text{rof}} - L_{\text{opt}}(d_{\alpha,\beta}) \tag{5.1}$$

S_1 denotes the strength of the initial signal returned to SD-RoF edge β. Subsequently, S_1 is determined as follows:

$$S_1 = S_0 - 2\Lambda(d_{\alpha,\beta})$$

Furthermore, $\Lambda(d)$ is the attenuation for distance d [m] between the SD-RoF edges.

$$\Lambda(d) = L_{\text{ant}} + C - A_{\text{rof}} + L_{\text{opt}}(d) \tag{5.2}$$

Moreover, if S_n denotes the strength of the signal that loops n-times, then S_n is generally determined as follows:

$$S_{\text{n}} = S_{n-1} - 2\Lambda(d_{\alpha,\beta}) \qquad \text{if } n > 0$$

Furthermore, if $N_{\beta,\text{ant}}$ denotes the initial noise-signal strength attained at the transmission antenna of base β from base α; then $N_{\beta,\text{ant}}$ is determined as follows:

$$N_{\beta,\text{ant}} = \max(N + A_{\text{rof}} + \text{nF}_{\text{rof}}, \Sigma_{i=1}^{\infty} S_i)$$

If $\Lambda(d_{\alpha,\beta}) \leq 0$, it is impossible to communicate because the self-interference wave continues to amplify. Therefore, assuming $\Lambda(d_{\alpha,\beta}) > 0$, the following equation is satisfied.

$$S_{n+1} < S_n \qquad \text{if } n \geq 0$$

Furthermore, assuming that $S_{n+1} \ll S_n$, $N_{\beta,\text{ant}}$ is determined by the following equation:

$$N_{\beta,\text{ant}} = \max(N + A_{\text{rof}} + \text{nF}_{\text{rof}}, S_1) \tag{5.3}$$

Hence, from equation (5.1) and equation (5.3), SNR_β is determined using the following equation:

$$\text{SNR}_\beta = S_0 - L_{\text{rf}}(d_\beta) - \max(N, N_{\beta,\text{ant}} - L_{\text{rf}}(d_\beta)) \tag{5.4}$$

Round Trip Time

To describe the latency, we evaluated the RTT of elastic bidirectional passthrough. We considered D [bytes] as the size of transmitted packets, B_{radio} [bps] as the communication speed of the terminal,

and c_{rof} [m/s] as the propagation speed in the optical fibers. In the proposed method, latency denotes the total time of propagation delay of the optical fiber (T_{fiber}), transmission delay in the wireless section (T_{radio}), processing delay (T_{proc}), and congestion delay (T_{con}). Incidentally, the transmission delay is absent in the wired section because the RoF is used for analog transmission. The analysis ignores the time to pass through wireless communication and electronic circuits. Here, T_{proc} corresponds to a constant because it is generally smaller than the propagation and transmission delays, and T_{con} is a constant because it is a temporary factor based on the status of the transmission line.

If T_{fiber} denotes the time taken for a data packet to pass through $d_{\alpha,\beta}$ from terminal α to terminal β, then T_{fiber} is determined by the following equation:

$$T_{\mathrm{fiber}} = \frac{d_{\alpha,\beta}}{c_{\mathrm{fiber}}} \tag{5.5}$$

In addition, T_{radio} is determined by the following equation.

$$T_{\mathrm{radio}} = \frac{8D}{B_{\mathrm{radio}}} \tag{5.6}$$

Therefore, $T_{\mathrm{delay,proposed}}$ denotes the time required for data to travel from terminal α to β via the proposed method. Hence, $T_{\mathrm{delay,proposed}}$ is determined by the following equation:

$$T_{\mathrm{delay,proposed}} = T_{\mathrm{fiber}} + 2T_{\mathrm{radio}} + T_{\mathrm{proc}} + T_{\mathrm{con}} \tag{5.7}$$

Hence, if $T_{\mathrm{rtt,proposed}}$ denotes the RTT taken to communicate between terminals α and β via proposed method, $T_{\mathrm{rtt,proposed}}$ can be determined by the following equation:

$$\begin{aligned} T_{\mathrm{rtt,proposed}} &= 2T_{\mathrm{delay,proposed}} \\ &= \frac{2d_{\alpha,\beta}}{c_{\mathrm{fiber}}} + \frac{32D}{B_{\mathrm{radio}}} + 2T_{\mathrm{proc}} + 2T_{\mathrm{con}} \end{aligned} \tag{5.8}$$

5.3.2 Cost Estimation by Prototype Implementation

In this evaluation, we evaluate the estimated cost of prototype implementation. During actual operation, costs must be determined based on the number of users, the range of applicable applications, the need to replace the existing network.

Elastic RoF Module

We implemented an Elastic RoF Module using four frequency converters, one switch matrix, and four signal synthesizers. Table 5.1 lists the total cost of the parts used, which amounted to approximately $995. The components of this module include the STM32F446RET6 (functioning as the control command decoder) and MAX2830ETM (used as LNA, demodulator, LPF, VGA, modulator, and PA). The switch matrix comprised Anaren PD2328J5050S2HFs and Panasonic Electric Works ARS1412s.

Table 5.1: Parts of elastic RoF module

parts	unit cost [$]	quantity	cost [$]
STM32F446RET6	8.45	5	42.25
MAX2830ETM	9.26	8	74.08
TLV3201AIDBVR	1.20	16	19.20
LTC2630CSC6-HZ10	3.53	16	56.48
ARS1412	6.77	48	324.96
PD2328J5050S2HF	0.75	8	6.00
other parts	–	–	471.73
total			994.70

Table 5.2: Parts of analog cancellator

parts	unit cost [$]	quantity	cost [$]
SKY12343-364LF	6.89	4	27.56
MAPS-010164	71.41	4	285.64
CC2531F128	8.09	1	8.09
C8051F360	11.89	1	11.89
other parts	–	–	18.35
total			351.53

This implementation demonstrates the feasibility of constructing SD-RoF networks using relatively inexpensive devices compared to those utilized in elastic optical networks and recent core networks. It suggests that it is possible to expand communication areas and enhance network flexibility while keeping infrastructure costs low.

Analog Cancellator

We implemented an analog cancellator using four attenuators, four-phase shifters, one transmitting circuit, and one controller. This analog cancellator reduces self-interference within the SD-RoF network, improving signal quality.

Table 5.2 lists the components used in the implementation. The total cost of implementation was approximately $351.53. In constructing the analog cancellator, we used Skyworks SKY12343-364LF as attenuators and M/A-COM Technology Solutions Inc. MAPS-010164 as phase shifters. The transmitting circuit was built using Texas Instruments CC2531F128, and Silicon Labs C8051F360 was employed as the controller.

5.3.3 Experimental Evaluation by Prototype Implementation

Elastic RoF Module

Table 5.3(a), (b) shows the essential characteristics of the Elastic RoF module. The noise figure of the frequency converter is approximately 15 dB, and the frequency switching delay is typically 30 μs due to the reset time of the MAX2830ETM. The insertion loss of the switch matrix and signal

Table 5.3: Basic characteristics of elastic RoF module

(a) Frequency converter

Characteristics	Value
Noise figure [dB]	15
Frequency switching delay [μs]	typ. 30

(b) Switch matrix and signal synthesizer

Characteristics	Value
Insertion loss [dB]	10.4
Average isolation[dB]	48.6
Switching delay [ms]	max. 10

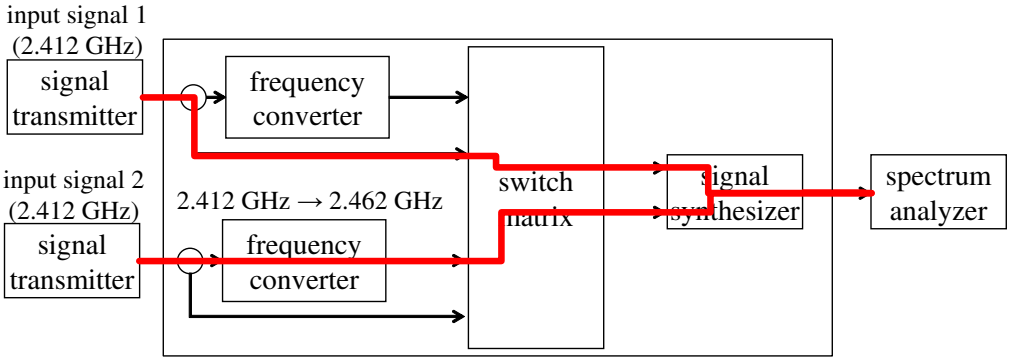
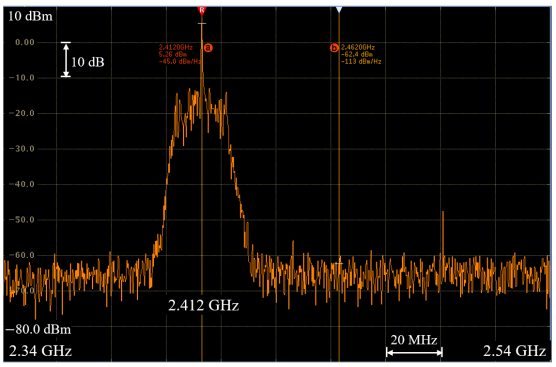


Figure 5.12: Experiment environment of elastic RoF module

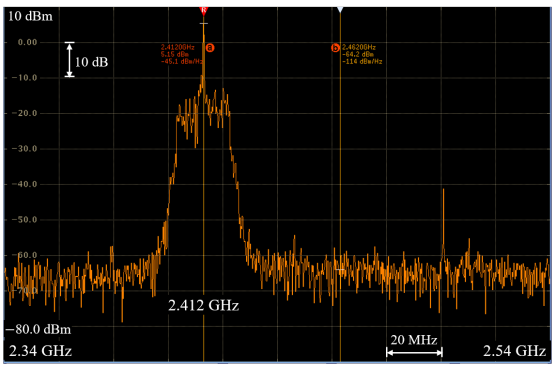
synthesizer is about 10.4 dB, and the average isolation for each input/output port is approximately 48.6 dB. This insertion loss includes the distribution loss of signals in the signal synthesizers and switch matrix. Average isolation indicates that unwanted input signals are added to the output signal as 48.6 dB lower power noise. The switching delay for the switch matrix configuration change is expected to be within 10 ms due to the switching time of the Panasonic Electric Works ARS1412.

Figure 5.12 depicts the experimental environment. Each signal transmitter generated a 2.412 GHz OFDM signal with a 20 MHz bandwidth. The Elastic RoF module received these signals, performed frequency conversion on one of them, and then multiplexed them. The output signal from the Elastic RoF module was fed into a spectrum analyzer.

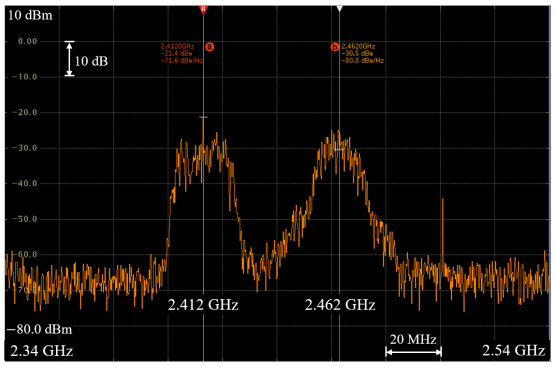
Figure 5.13 shows the results of frequency conversion and multiplexing using the Elastic RoF module. Figures 5.13(a) and (b) display two 2.412 GHz OFDM signal waveforms as input to the Elastic RoF module. The input signal 1 shown in Figure 5.13(a) was not frequency-converted, while the input signal 2 shown in Figure 5.13(b) was frequency-converted from 2.412 GHz to 2.462 GHz. Figure 5.13(c) presents the signal waveform of the output signal from the Elastic RoF



(a) Input OFDM signal 1



(b) Input OFDM signal 2 (before frequency-converting from 2.412 GHz to 2.462 GHz)



(c) Output signal

Figure 5.13: Frequency conversion and multiplexing of OFDM signals using elastic RoF Module

module. It confirms that input signal 1 without frequency conversion and input signal 2 with frequency conversion from 2.412 GHz to 2.462 GHz were multiplexed. The signal synthesizer and switch matrix reduced the power of both input signals 1 and 2. However, the bandwidth of input signal 2 increased compared to the original signal due to noise from the frequency converter. To avoid interference of electrical signals in multiplexing, it is necessary to properly determine the conversion frequency due to the increased occupied bandwidth after frequency conversion.

Analog Cancellator

To verify the cancellation performance of the analog cancellator, we measured the Received Signal Strength Indicator (RSSI) of signals before and after self-interference cancellation using a spectrum analyzer. We compared the cancellation performance for an unmodulated continuous wave and IEEE 802.15.4 data packets.

Figure 5.14 depicts the experimental environment for evaluating the self-interference cancellation performance of the analog cancellator. The transmitted signal was distributed to the analog cancellator and an attenuator. The attenuator simulated a 10 dB propagation loss between the transmitting and receiving antennas of the same SD-RoF edge. The signal reproduced the self-

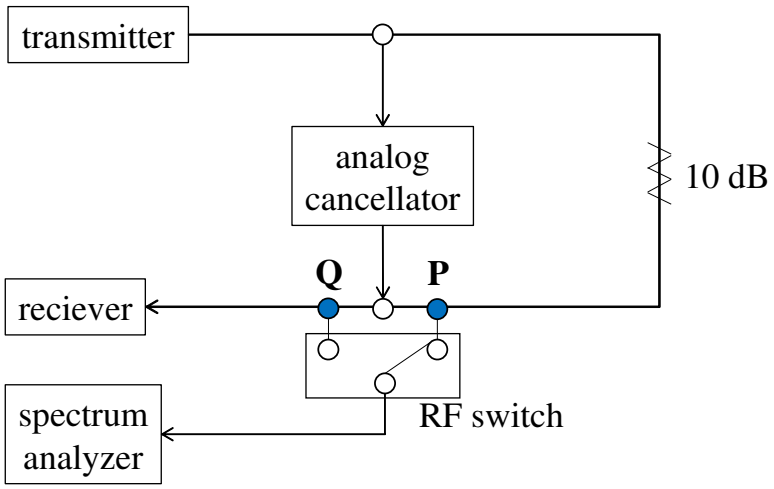


Figure 5.14: Evaluation environment for the self-interference cancellation performance of the analog cancellator

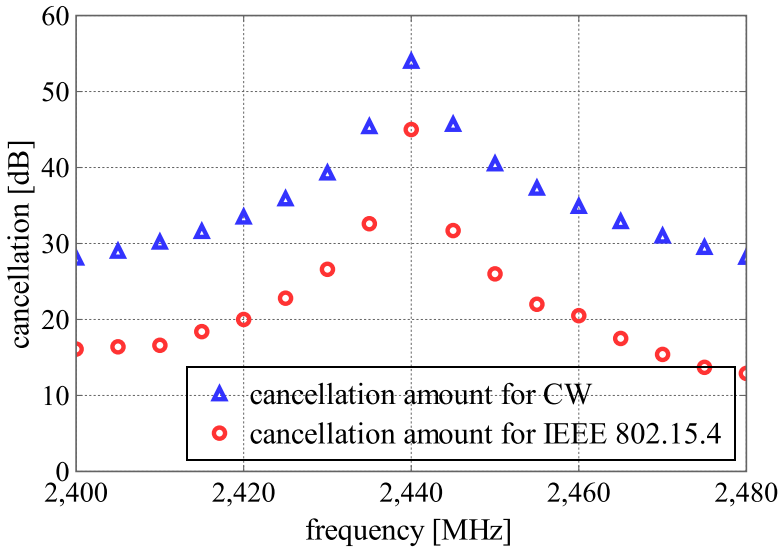


Figure 5.15: Measured cancellation performance of analog cancellator

interference signal passed through the attenuator, and the frequency of the unmodulated continuous wave input from the transmitter was varied in the range of 2.400 to 2.480 GHz. The amplification factor A_{rof} of RoF is 3 dB. The RSSI was measured at points P and Q as shown in Figure 5.14, and the cancellation amount was calculated by comparing the RSSI at these points.

Figure 5.15 shows the cancellation amount for unmodulated continuous waves and IEEE 802.15.4 data packets at a center frequency of 2.400 GHz. The horizontal axis represents the frequency of the input signal, while the vertical axis shows the cancellation amount. The results indicate that a maximum cancellation of approximately 54 dB was achieved for unmodulated continuous waves

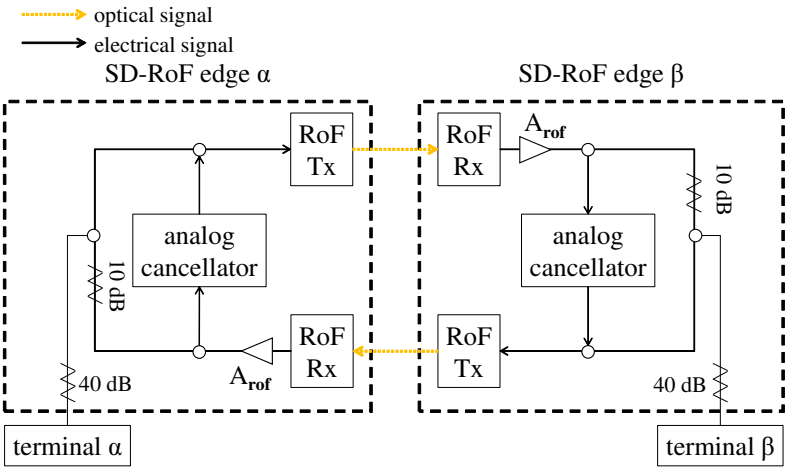


Figure 5.16: Evaluation environment with analog cancellator

and a maximum of approximately 45 dB for IEEE 802.15.4 data packets. The bandwidth of IEEE 802.15.4 data packets is 2 MHz, which may explain the lower cancellation performance compared to unmodulated continuous waves. These experimental results demonstrate the potential of the proposed analog cancellator to reduce self-interference and effectively enhance signal quality in SD-RoF networks.

5.3.4 Demonstration of Elastic Bidirectional Passthrough

We evaluated elastic bidirectional passthrough in three ways: Packet Error Rate (PER), RTT, and protocol applicability.

In PER and RTT evaluation, the transmitting circuit was composed of CC2531F128. CC2531F128 is a wireless transmission/reception module that includes the function of transmitting unmodulated continuous waves and the function of transmitting IEEE 802.15.4 data packets. The transmission frequency corresponded to 2.440 GHz, and the transmission power corresponded to 0 dBm. The SD-RoF edge was composed of the transmission antenna, reception antenna, and analog cancellator. The RoF transmitter and receiver used ET-615 and ER-615, respectively. The operation was performed via the FTDI Virtual COM Port Driver. Terminals α and β were placed in the shield box and shielded tent to prevent direct wireless communication between terminals α and β, respectively. The shield box used a shield cube of 45 cm ($450 \times 450 \times 450$ [mm]) made by TOYO Corporation. The shielding performance approximately corresponded to the measured 60 dB. The shield tent used the EMI shield tent ($2400 \times 4800 \times 2000$ [mm]) made by TEIESJAPAN Corporation. The shielding performance approximately corresponded to the measured 50 dB.

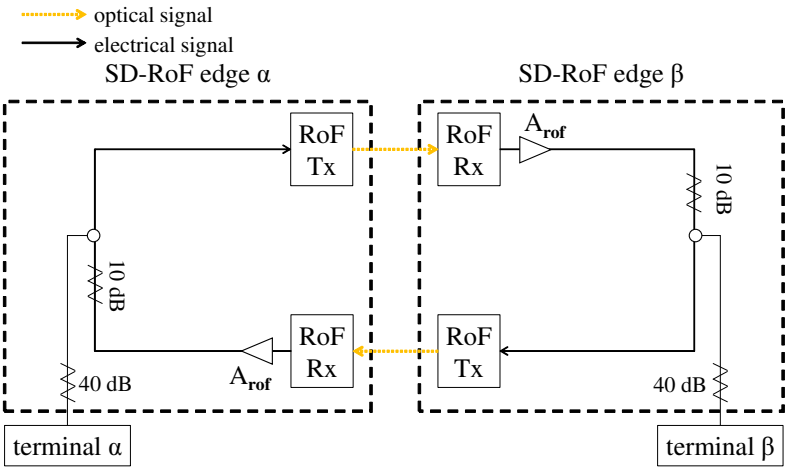


Figure 5.17: Evaluation environment without analog cancellator

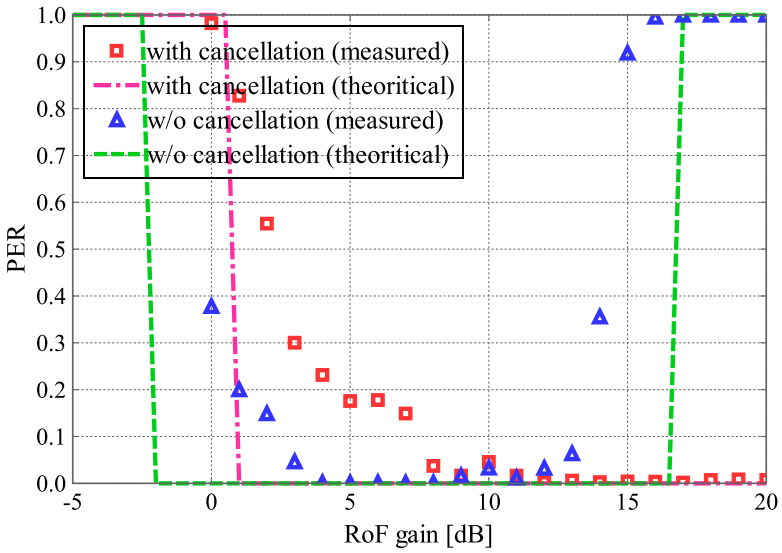


Figure 5.18: Comparison of PER with and without the analog cancellator

PER

To evaluate the communication feasibility of wireless access networks, we measured the PER. Specifically, we measured the PER of packet transmission from terminal α to terminal β while varying the amplification value in the RoF (A_{rof}), comparing the results with and without the amplitude and insertion of the delay control circuit. Increasing the value of A_{rof} is desirable to extend the communication distance of each SD-RoF edge.

Figure 5.16 presents the evaluation environment using the analog cancellator, while Figure 5.17 presents the environment without the analog cancellator. The evaluation involved connecting

the terminals and SD-RoF edges using a coaxial cable. Attenuators were inserted between the transmitting and receiving antennas of the SD-RoF edges to simulate self-interference at each base station, simulating approximately 10 dB of attenuation between antennas. Furthermore, the terminal transmitter and the receiving antenna of the SD-RoF edge were connected using an attenuator to replicate the propagation loss between them, approximately 40 dB. The SD-RoF edges were linked to 20 km long optical fibers.

In this evaluation, the parameters used were as follows:

- P_{tx} [dBm] = 0 dBm
- $L_{\mathrm{rf}}(d_\alpha)$ and $L_{\mathrm{rf}}(d_\beta)$ [dB] = 43 dB
- NF_{rof} [dB] = 17 dB
- $d_{\alpha,\beta}$ [km] = 20 km
- $L_{\mathrm{opt}}(d_{\alpha,\beta})$ [dB] = 4 dB
- L_{ant} [dB] = 15 dB
- N [dB] = −97 dB

We assumed 0.2 dB/km as the attenuation in the optical fiber and considered the attenuation at the splitter. Based on equation (5.4), the SNR_β in this environment is calculated as follows:

$$\mathrm{SNR}_\beta = \begin{cases} A_{\mathrm{rof}} + 7 & \text{if } A_{\mathrm{rof}} < \frac{31+2C}{3} \\ 38 + 2C - 2A_{\mathrm{rof}} & \text{else} \end{cases} \tag{5.9}$$

The required SNR for IEEE 802.15.4 is at least 5 dB. With C [dB] being 0 dB without the analog cancellator, the conditions for feasible communication are determined from equation (5.9) as follows:

$$-2 \leq A_{\mathrm{rof}} \leq 16.5 \qquad \text{if } C = 0, \tag{5.10}$$

Conversely, with C [dB] being 45 dB with the analog cancellator, the conditions are:

$$1 \leq A_{\mathrm{rof}} \leq 61.5 \qquad \text{if } C = 45 \tag{5.11}$$

From Equations 5.10 and 5.11, communication is feasible only when using the analog cancellator for $16.5 < A_{\mathrm{rof}} \leq 61.5$.

Figure 5.18 compares the PER with and without the analog cancellator. The measured results indicate that the case without the analog cancellator has a lower PER and better communication quality when A_{rof} is less than 8 dB. Inserting an analog cancellator results in signal attenuation due to the splitter, leading to packet errors. Conversely, communication quality with the circuit

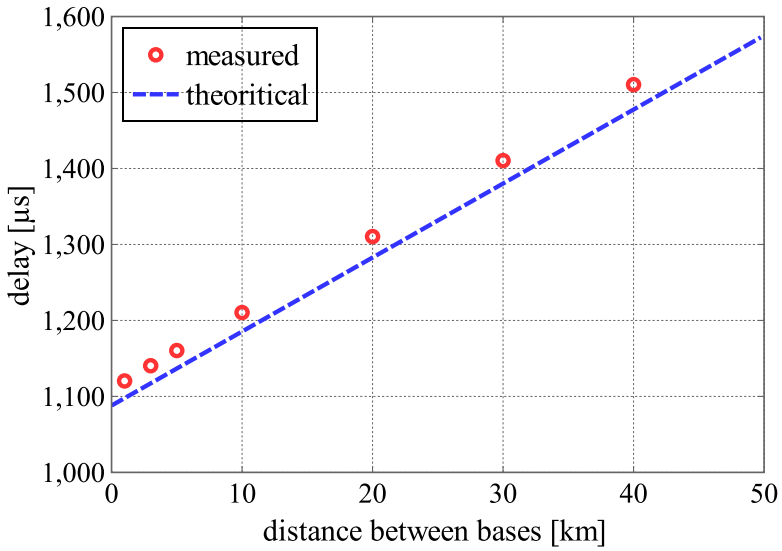


Figure 5.19: Comparison of the measured RTT and theoretical RTT derived from equation (5.8)

surpasses that without the circuit when the amplification value at the RoF is 12 dB or more. Inserting the analog cancellator allows canceling self-interfering signals due to increased amplification in RoF and communication.

The results in Figure 5.18 suggest that the outcomes are close to theoretical analysis. However, the communication feasibility conditions are narrower in the measured results compared to the theoretical ones, likely due to the nonlinear distortion and noise sensitivity of the analog RoF.

RTT

We measured the RTT to evaluate the proposed method's latency characteristics. The RTT evaluation environment was the same as the previously mentioned PER evaluation environment (see Figure 5.16). In this experiment, the RTT represented when terminal α began transmitting an IEEE 802.15.4 data packet to when it completed receiving the Acknowledgment (ACK) packets from terminal β. Terminals and SD-RoF edges were wirelessly connected. Terminal α sent IEEE 802.15.4 data packets to terminal β, which, upon receiving these packets, sent back ACK packets to terminal α. The data packet size corresponded to 44 bytes, and the ACK packet size corresponded to 24 bytes. Terminal α retransmitted if the ACK packets from terminal β were not received within 1 ms. The maximum number of retransmissions and Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) attempts were set to three and five times, respectively, following IEEE 802.15.4 standards. Time measurement was conducted using the 16-bit timer at 1.6MHz in the CC2531F128.

Figure 5.19 shows a comparison between the measured RTT and the theoretical RTT de-

	1 m	200 m	600 m	1000 m	3000 m	5000 m	10000 m	20000 m
IEEE 802.15.4	✓	✓	✓	✓	✓	✓	✓	✓
IEEE 802.11bg	✓	✓	✓	✓	✓			
BR/EDR	✓	✓	✓	✓				
BLE	✓	✓	✓	✓				

Table 5.4: The protocol applicability for SD-RoF

rived from Equation (5.8). The measured RTTs were approximately 1,120 μs, 1,140 μs, 1,160 μs, 1,110 μs, 1,310 μs, 1,410 μs, and 1,510 μs for fiber lengths of 1 km, 3 km, 5 km, 10 km, 20 km, 30 km, and 40 km, respectively. In the theoretical value calculations, the light speed in optical fiber c_{rof} was approximately 2.053×10^8 [m/s], the radio bandwidth B_{radio} was 250 kbps, and the data size D was 17 bytes. As shown in Figure 5.19, a correlation between the measured and theoretical results was observed. This confirms that, due to the elimination of processing delays in the communication path by using analog transmission, it is possible to reduce the Round-Trip Time (RTT) to approach the physical limit of propagation delay. The proposed method has been demonstrated to achieve low-latency communication of less than 1 ms in one direction for distances up to 40 km, closely aligning with the theoretical limits of propagation delay.

Protocol applicability

To evaluate the applicability of communication protocols to the proposed method, we investigated the feasibility of communication using several protocols. Specifically, we conducted experiments with various lengths of optical fiber using 2.4 GHz communication protocols (IEEE 802.11bg, IEEE 802.15.4, Bluetooth BR/EDR, and Bluetooth BLE) to determine the range of direct communication possible via SD-RoF. The experiments were conducted using single-core optical fiber and 2.4 GHz antennas in a radio anechoic chamber.

The lengths of optical fibers used were 1 m, 200 m, 600 m, 1,000 m, 3,000 m, 5,000 m, 10,000 m, and 20,000 m. For IEEE 802.15.4 communication, we used the terminals described in Section 5.3.4. For IEEE 802.11bg communication, Buffalo's WZR-300HP and WZR-600DHP were used. Two PCs were used as Bluetooth BR/EDR communication terminals for file transfer. We confirmed the connection between a smartphone and a smartwatch for Bluetooth BLE communication.

The results are shown in Table 5.4. IEEE 802.15.4 could communicate over all distances, but at 20,000 m, retransmissions frequently occurred. On the other hand, IEEE 802.11bg communication was only successful up to 3,000 m, likely due to the failure of CSMA/CA to function correctly over long distances. BR/EDR and BLE could communicate up to 1,000 m, but communication became unstable beyond that distance due to master-slave time synchronization issues. Specifically, the Bluetooth time synchronization protocol tolerates a gap of ± 10 μs, and the use of optical fiber over 1,000 m results in a round-trip delay exceeding 10 μs, making communication difficult.

These results indicate that while IEEE 802.15.4 applies to SD-RoF, other protocols have spe-

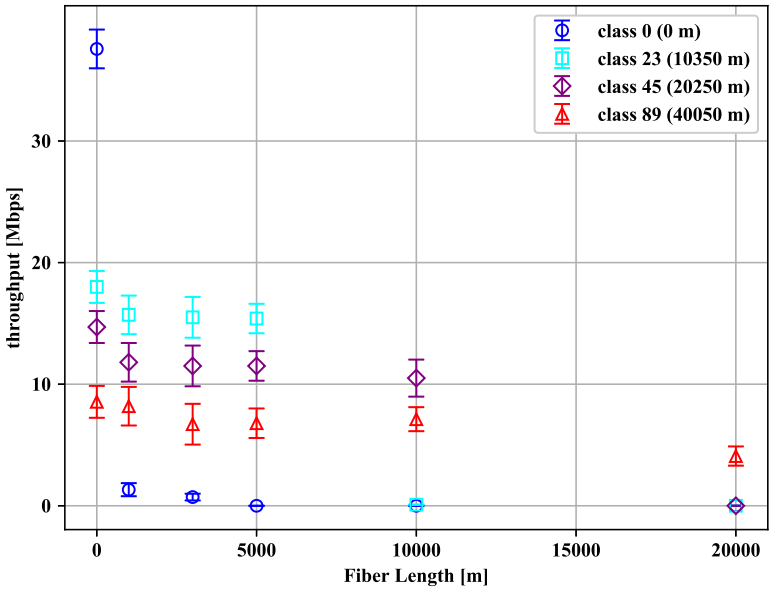


Figure 5.20: The effect of coverage class value

cific limitations. In particular, the effects of CSMA/CA and Bluetooth time synchronization issues become significant over long distances, necessitating additional measures to overcome these problems.

In particular, the effects of CSMA/CA and Bluetooth time synchronization issues become significant over long distances, necessitating additional measures to overcome these problems.

Protocol optimization

In order to discuss the optimization of existing communication protocols to SD-RoF, we conducted an experiment to extend IEEE 802.11bg. Specifically, we conducted experiments to modify the spatial propagation delay defined in IEEE 802.11bg. The optical fibers' lengths were 1 m, 1,000 m, 3,000 m, 5,000 m, 10,000 m, and 20,000 m. In IEEE 802.11, the space propagation delay is less than 1 ms, since direct communication over long distances is not usually used. However, when direct communication is performed via SD-RoF, a propagation delay of 5 μs per 1 km occurs, and collision avoidance by CSMA/CA does not work properly. The issue of CSMA/CA in communication over RoF has been discussed for some time. In the document [115], it is proposed that CSMA/CA can work properly by changing the slot time. This method suffers from a sharp drop in throughput during long-distance communication. In contrast, the paper [116] proposes to operate CSMA/CA normally by adjusting NAV. Compared to the method of changing the slot time, this method can reduce the throughput degradation because it can minimize the time extension.

Equation 5.12 shows the normal slot time definition in IEEE 802.11bg.

$$\begin{aligned} aSlotTime &= aCCATime + aRxTxTurnaroundTime + aAirPropagationTime \\ &\quad + aMACProcessingDelay \\ &= 9\ [\mu s] \quad (aAirPropagationTime << 1), \end{aligned} \tag{5.12}$$

In SD-RoF, it is necessary to consider the propagation delay of the optical fiber ($aRoFPropagationTime$). If the propagation delay due to the optical fiber is 5μ [s/km], it can be expressed as follows when the length of the optical fiber is a [km].

$$aRoFPropagationTime = 5a \tag{5.13}$$

Equation 5.14 shows the slot time required for application to the proposed system.

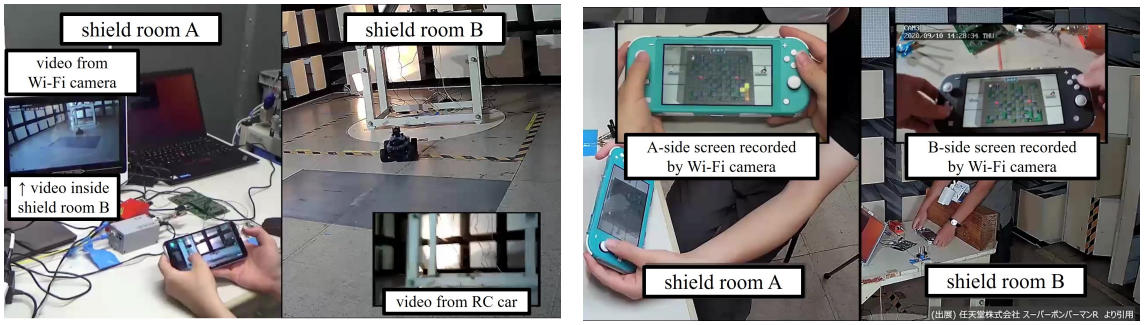
$$\begin{aligned} aSlotTime_{SD-RoF} &= aCCATime + aRxTxTurnaroundTime + aAirPropagationTime \\ &\quad + aMACProcessingDelay + aRoFPropagationTime \\ &= aSlotTime + aRoFPropagationTime \\ &= 2 \times (4.5 + 5a)\ [\mu s] \quad (aAirPropagationTime << 1), \end{aligned} \tag{5.14}$$

The space propagation delay is defined in the IEEE 802.15.4 standard, and the maximum space propagation delay can be set to 114,750 m. In this experiment, the slot time was changed by setting the spatial propagation delay equal to the length of the optical fiber, and the actual experiment was conducted. The experiment was run on Buffalo's WZR-300HP and WZR-600DHP with OpenWrt installed.

Figure 5.20 shows the results. Figure 5.20 shows that by setting the optimal space propagation delay for each optical fiber length, the existing protocols can communicate over distances where communication was impossible in Chapter . The maximum throughput for each optical fiber length is achieved when the minimum class that can set a space propagation delay longer than the optical fiber length is set. On the other hand, as pointed out in the literature [116], the throughput decreased significantly. By setting a large class, it is possible to realize communication without changing the existing protocol in any optical fiber length. However, to improve the throughput, it will be necessary to set optimal parameters and develop new protocols.

Demonstration

In order to confirm the actual behavior of communication through the proposed system, we conducted experiments using several real applications. The experiments were conducted in two shield rooms to prevent the terminals from communicating directly. The optical fiber used in the experiments was about 10 m.



(a) Remote control of radio controlled model car (b) Remote communication of local-multiplayer game

Figure 5.21: Frequency conversion and multiplexing of OFDM signals using elastic RoF Module

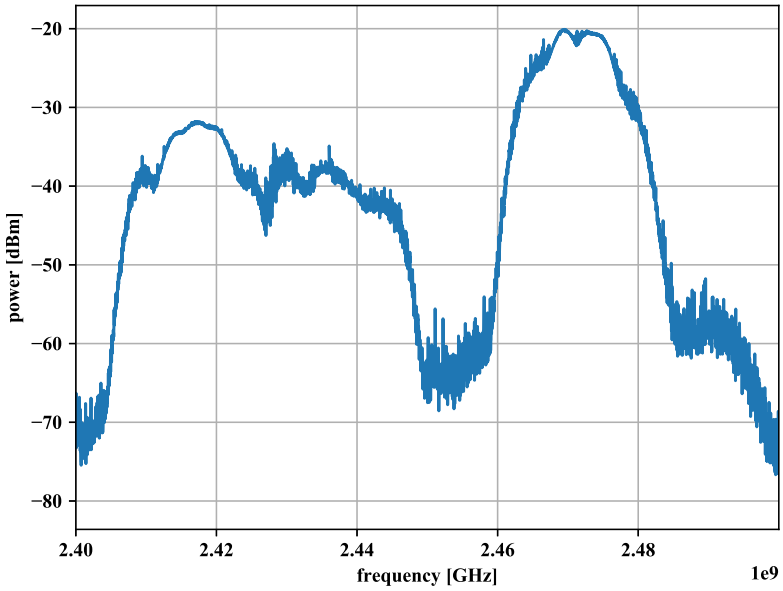


Figure 5.22: Waveform during the experiment

In order to demonstrate the feasibility of remote operation via the proposed system, we experimented using a radio-controlled car controlled by a smartphone. In Fig 5.21(a), the smartphone in Shield Room A controlled the radio controller in Shield Room B. The camera in the radio-controlled car in Shield Room B transmits images to the smartphone in Shield Room A. Using the proposed system, the experimenter in Shield Room A can control the radio controller in Shield Room B based on the image output from the smartphone.

In order to show the expansion of the radio space by the proposed system, we conducted an experiment using a radio controller controlled by a smartphone. In Fig 5.21(b), we performed an experiment using local communication between a game console in Shield Room A and a game console in Shield Room B. We used Nintendo Switch as the game console and Bomberman as the

game. We used a Nintendo Switch as a game console and Bomberman as a game. Using the proposed system, we can realize a local communication game between the game machines in the same space, but they are far away. This paper shows that the proposed system can be used for local communication games.

In both experiments (Fig 5.21(a) and Fig 5.21(b)), the video is always transmitted from the monitoring camera in Shield Room B to the monitor in Shield Room A. In the experiment in Fig 5.21(a) and Fig 5.21(b), the video is always transmitted from the monitoring camera in Shield Room B to the monitor in Shield Room A. In order to achieve the wideband capability, which is one of the requirements of the proposed system, it is necessary to be able to broadcast multiple communications simultaneously. Fig 5.22 shows the waveform of the experiment. In Fig 5.22, ch1, ch6, and ch11 are used to communicate with a game, a radio-controlled car, and a surveillance camera, respectively. In the experiment, we were able to perform these communications simultaneously. Therefore, it is shown that the proposed system has a high bandwidth capability that can be used for multiple applications simultaneously.

5.4 Conclusion

We have introduced SD-RoF, an architecture that seamlessly integrates optical and radio technologies. Central to this architecture are two pivotal functions: elastic wireless service and elastic bidirectional passthrough. Our work encompasses the design, prototype implementation of SD-RoF, and a preliminary evaluation of the implemented circuit. Additionally, we have conducted demonstration experiments focusing on the elastic bidirectional passthrough.

Our findings reveal that devices utilizing short-range wireless communication protocols, such as IEEE 802.15.4 and IEEE 802.11, can achieve communication over several kilometers using SD-RoF. However, the feasible communication distance for each protocol varies due to the propagation delay in fiber. Addressing this, it is essential to either operate within the acceptable area or optimize parameters to extend the communication range, ensuring the effective use of existing protocols with SD-RoF.

This network architecture is poised to revolutionize the mobile cloud gaming experience, addressing critical latency and network traffic management challenges. Our research stands out by tackling the theoretical and practical aspects of deploying cloud gaming systems on mobile networks. It significantly advances the current knowledge and capabilities in cloud gaming and lays a robust foundation for future innovations in mobile network technology.

Chapter 6

Conclusion

In this paper, we focused on unraveling and addressing one of the most difficult challenges in cloud gaming: the issue of response latency. The delay between a player's action and the game's response is a critical factor that can significantly impact the gaming experience, particularly in fast-paced and interactive genres. Our research delves into the potential of speculative execution. Speculative execution is a cutting-edge approach designed to revolutionize cloud gaming by preempting player actions and preemptively rendering game scenarios. This comprehensive study examines the technical feasibility of a speculative cloud gaming system and evaluates its efficiency, scalability, and adaptability across various gaming genres. Our work has set a new benchmark in pursuing seamless and immersive cloud gaming experiences by bridging theoretical concepts with practical implementations.

Throughout our research, we have systematically tackled the multifaceted aspects of speculative execution, ranging from reducing computational cost and network traffic to optimizing video transmission techniques and combining the pioneering development of a novel network architecture tailored for mobile cloud gaming systems. With current computing power, it will be difficult to achieve the performance required for speculative execution in cloud gaming systems after all. However, this result is not entirely negative. The proposed method makes it possible to reduce speculative execution and delay under certain conditions with a realistic amount of computation. This is an important advance in research on cloud gaming systems, as it demonstrates the effectiveness of speculative execution in future. Each chapter of this paper contributes collectively to advancing our understanding of how to effectively conceal response latency and enhance the overall quality of cloud gaming.

In Chapter 3, we developed a pattern reduction method to address the significant computational cost and network traffic challenges associated with speculative execution. Our method leverages empirical data analysis, utilizing actual gaming data for validation. The results demonstrated a substantial reduction in the number of frames that needed rendering and a marked decrease in network traffic. This achievement not only proved the viability of speculative execution within

realistic computational costs but also opened new pathways for its application in cloud gaming systems, potentially transforming user experiences by reducing perceptible delays.

Chapter 4 marked a significant breakthrough in data transmission for cloud gaming. We introduced a novel video transmission method for the dynamic nature of speculative execution. This innovative approach transcended the limitations of conventional video compression technologies, facilitating efficient transmission of multiple potential game scenarios. The method's flexibility to adapt to the rapid changes inherent in gaming scenarios makes it a cornerstone for future speculative cloud gaming technologies, enabling smoother and more responsive gameplay even in bandwidth-intensive situations.

In Chapter 5, We proposed a revolutionary network architecture that employs RoF technology, targeting cloud gaming systems on mobile networks. This architecture significantly reduces processing delays and effectively manages increased traffic volumes, enhancing the overall gaming experience on mobile platforms. This innovative approach contributes to reducing latency in mobile cloud gaming and paves the way for the broader adoption of cloud gaming technology in mobile networks. The implications of this technology extend beyond gaming, offering potential advancements in various mobile network-based services requiring low latency and high throughput.

Overall, our research substantially contributes to the cloud gaming field. We have successfully navigated the challenges of implementing speculative execution, addressing critical issues such as computational overheads, network traffic management, and real-time data transmission. Our findings demonstrate how these technological advancements can significantly elevate the gaming experience, bringing it closer to real-time interaction.

Future research directions are multifaceted. They involve further optimizing and refining our developed technologies, testing their performance and scalability under various network conditions, and exploring their practical deployment in real-world scenarios. When operating a cloud gaming system, adaptation to the market and sustainability of service provision are important factors. Implementation of the proposed method evaluates the balance between operational costs and technology investment, and becomes a new option for strategies for sustainable service provision. Specifically, operators will be able to consider not only optimizing server placement according to market demand using conventional edge computing technology, but also consolidating servers using low-latency technology.

Furthermore, the most critical issue in the practical application of the proposed method is the implementation of the game engine and the associated processing load. In order to reduce the processing load, we believe that reducing the number of patterns performed in this research is effective. Reducing the number of patterns in speculative execution is important not only for reducing traffic, but also for fundamentally resolving various issues related to speculative execution. On the other hand, when it comes to implementing a game engine, although it is possible to

conceptualize it theoretically, it is considered difficult to realize it with the current computing power, considering the required memory usage and processing speed. In the future, we believe that an approach that shares game assets between each pattern will be effective in order to efficiently process multiple patterns. Although this approach does not directly contribute to faster processing, it can significantly reduce memory usage.

Additionally, the insights gained from our study are valuable for applications beyond cloud gaming. They have the potential to influence other areas reliant on mobile networks, leading to breakthroughs in digital communication and interactive technologies. For example, Katsuyama et al. [117] proposed a speculative execution method in robot operations. The applicability of speculative execution is not limited to cloud gaming systems, but may also extend to remote control of robots, remote driving, etc. Implementing speculative execution in real-world scenarios may be on the horizon. By using SD-RoF, it is possible to envision the construction of flexible networks that can accommodate speculative applications with various wireless communication standards. Building a network that can seamlessly accommodate advanced sensors and integrating them into speculative execution frameworks opens up exciting possibilities for augmented reality applications.

Acknowledgement

I would like to express my sincerely appreciation to continued support of my supervisors Specially Appointed Professor Takashi Watanabe of Osaka University through trials and tribulations of this Ph.D thesis. Again I express my heartiest gratitude to them for their encouragement and invaluable comments in preparing this thesis.

I am very grateful to Professor Masayuki Murata, Professor Toru Hasegawa, Professor Hirozumi Yamaguchi and Professor Hideyuki Shimonishi of Osaka University for their invaluable comments and helpful suggestions concerning this thesis.

I am heartily grateful to Associate Professor Shunsuke Saruwatari for the precious advises and technical discussions provided through out the research.

I would like to thank Assistant Professor Takuya Fujihashi for his valuable comments and disscussions.

I would also like to extend my thanks to Ami Takahashi for their invaluable assistance on laboratory's management and tasks.

Thanks go to everyone of Intelligent Networking laboratory for their feedback, encouragement and support.

Finally, I would like to thank my family and my friends for their help and understanding.

Bibliography

[1] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[2] C.-Y. Huang, K.-T. Chen, D. Chen, H.-J. Hsu, and C. Hsu, "Gaminganywhere: The first open source cloud gaming system," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 10, no. 1s, pp. 1–25, Jan. 2014.

[3] A. More, "Cloud gaming market growth (USD 47 billion by 2032 at 48.20CAGR) global analysis by market.us," https://www.linkedin.com/pulse/cloud-gaming-market-growth-usd-47-billion-2032-4820-cagr-aboli-more/, Jul. 2023, accessed: 24 November 2023.

[4] M. Kobayashi, R. Murakami, K. Kizaki, S. Saruwatari, and T. Watanabe, "Wireless full-duplex medium access control for enhancing energy efficiency," *IEEE Transactions on Green Communications and Networking*, vol. 2, no. 1, pp. 205–221, Mar. 2018.

[5] S. Choy, B. Wong, G. Simon, and C. Rosenberg, "The brewing storm in cloud gaming: A measurement study on cloud to end-user latency," in *2012 11th Annual Workshop on Network and Systems Support for Games (NetGames)*, Nov. 2012, pp. 1–6.

[6] S. Baek, J. Ahn, and D. KimBaek, "Future business model for mobile cloud gaming: the case of south korea and implications," *IEEE Communications Magazine*, vol. 61, no. 7, pp. 68–73, Feb. 2023.

[7] O. S. Peñaherrera-Pulla, C. Baena, S. Fortes, E. Baena, and R. Barco, "Measuring key quality indicators in cloud gaming: Framework and assessment over wireless networks," *Sensors*, vol. 21, no. 4, Feb. 2021.

[8] K. Raaen, R. Eg, and C. Griwodz, "Can gamers detect cloud delay?" in *2014 13th Annual Workshop on Network and Systems Support for Games*, Dec. 2014, pp. 1–3.

[9] M. Amiri, H. A. Osman, S. Shirmohammadi, and M. Abdallah, "Toward Delay-Efficient Game-Aware data centers for cloud gaming," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 12, no. 5s, Dec. 2016.

[10] R. Gouareb, V. Friderikos, and A.-H. Aghvami, "Virtual network functions routing and placement for edge cloud latency minimization," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2346–2357, Sep. 2018.

[11] "Parsec official webcite," https://parsec.app/, accessed: 18 November 2023.

[12] D. Wu, Z. Xue, and J. He, "iCloudAccess: Cost-Effective streaming of video games from the cloud with low latency," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 8, pp. 1405–1416, Aug. 2014.

[13] M. Amiri, K. P. S. Malik, H. A. Osman, and S. Shirmohammadi, "Game-aware resource manager for home gateways," in *2016 IEEE International Symposium on Multimedia (ISM)*, Dec. 2016, pp. 403–404.

[14] S.-Y. Lien, S.-C. Hung, K.-C. Chen, and Y.-C. Liang, "Ultra-low-latency ubiquitous connections in heterogeneous cloud radio access networks," *IEEE Wireless Communications*, vol. 22, no. 3, pp. 22–31, 2015.

[15] M. Basiri and A. Rasoolzadegan, "Delay-Aware resource provisioning for Cost-Efficient cloud gaming," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 4, pp. 972–983, Nov. 2016.

[16] X. Zhang, H. Chen, Y. Zhao, Z. Ma, Y. Xu, H. Huang, H. Yin, and D. O. Wu, "Improving cloud gaming experience through mobile edge computing," *IEEE Wireless Communications*, vol. 26, no. 4, pp. 178–183, Apr. 2019.

[17] M. Amiri, A. Sobhani, H. A. Osman, and S. Shirmohammadi, "SDN-enabled game-aware routing for cloud gaming datacenter network," *IEEE Access*, vol. 5, pp. 18 633–18 645, Sep. 2017.

[18] "NVIDIA corporation. NVIDIA geforce NOW," https://www.nvidia.com/en-us/geforce-now/system-reqs/, accessed: 29 Octorber 2022.

[19] NVIDIA, "Shackling jitter and perfecting ping, how to reduce latency in cloud gaming," https://blogs.nvidia.com/blog/what-is-latency-jitter-ping-cloud-gaming/, Mar. 2021, accessed: 18 November 2023.

[20] I. Burstein, "Nvidia data center processing unit (DPU) architecture," in *2021 IEEE Hot Chips 33 Symposium (HCS)*, Aug. 2021, pp. 1–20.

[21] M. Suznjevic, I. Slivar, and L. Skorin-Kapov, "Analysis and QoE evaluation of cloud gaming service adaptation under different network conditions: The case of NVIDIA geforce NOW," in *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, Jun. 2016, pp. 1–6.

[22] K.-T. Chen, Y.-C. Chang, H.-J. Hsu, D.-Y. Chen, C.-Y. Huang, and C.-H. Hsu, "On the quality of service of cloud gaming systems," *IEEE Transactions on Multimedia*, vol. 16, no. 2, pp. 480–495, Feb. 2014.

[23] H. Madiha, L. Lei, A. A. Laghari, and S. Karim, "Quality of experience and quality of service of gaming services in fog computing," in *Proceedings of the 2020 4th International Conference on Management Engineering, Software Engineering and Service Sciences*, ser. ICMSS 2020. Association for Computing Machinery, May 2020, pp. 225–228.

[24] Y. Lin and H. Shen, "Cloudfog: Leveraging fog to extend cloud gaming for thin-client MMOG with high quality of service," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 2, pp. 431–445, Feb. 2017.

[25] J. Søgaard, M. Shahid, J. Pokhrel, and K. Brunnström, "On subjective quality assessment of adaptive video streaming via crowdsourcing and laboratory based experiments," *Multimedia Tools and Applications*, vol. 76, pp. 16 727–16 748, Aug. 2017.

[26] J. Wu, C. Yuen, N.-M. Cheung, J. Chen, and C. W. Chen, "Enabling adaptive High-Frame-Rate video streaming in mobile cloud gaming applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 12, pp. 1988–2001, Dec. 2015.

[27] K. Pires and G. Simon, "DASH in twitch: Adaptive bitrate streaming in live game streaming platforms," in *Proceedings of the 2014 Workshop on Design, Quality and Deployment of Adaptive Video Streaming*, ser. VideoNext '14. Association for Computing Machinery, 2014, pp. 13–18.

[28] A. Alhilal, T. Braud, B. Han, and P. Hui, "Nebula: Reliable low-latency video transmission for mobile cloud gaming," in *Proceedings of the ACM Web Conference 2022*, ser. WWW '22, Apr. 2022, pp. 3407–3417.

[29] C. Gutterman, B. Fridman, T. Gilliland, Y. Hu, and G. Zussman, "Stallion: Video adaptation algorithm for Low-Latency video streaming," in *Proceedings of the 11th ACM Multimedia Systems Conference*, ser. MMSys '20. Association for Computing Machinery, May 2020, pp. 327–332.

[30] I. Slivar, L. Skorin-Kapov, and M. Suznjevic, "QoE-Aware resource allocation for multiple cloud gaming users sharing a bottleneck link," in *2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, Feb. 2019, pp. 118–123.

[31] R. Gharsallaoui, M. Hamdi, and T.-H. Kim, "A novel adaptive streaming approach for cloud-based mobile video games," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, Jun. 2017, pp. 1072–1077.

[32] S. Mori, Y. Mizoguchi, and M. Bandai, "An HTTP adaptive streaming method considering motion intensity," in *2018 IEEE 7th International Conference on Cloud Networking (CloudNet)*, Oct. 2018, pp. 1–3.

[33] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. K. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, Aug. 2003.

[34] J.-R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC)," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1669–1684, Oct. 2012.

[35] I. Storch, D. Palomino, B. Zatt, and L. Agostini, "Speedup evaluation of HEVC parallel video coding using tiles," *Journal of Real-Time Image Processing*, vol. 17, no. 5, pp. 1469–1486, Oct. 2020.

[36] S.-P. Chuah, N.-M. Cheung, and C. Yuen, "Layered coding for mobile cloud gaming using scalable Blinn-Phong lighting," *IEEE Transactions on Image Processing*, vol. 25, no. 7, pp. 3112–3125, Jul. 2016.

[37] Y. Xu, Q. Shen, X. Li, and Z. Ma, "A Cost-Efficient cloud gaming system at scale," *IEEE Network*, vol. 32, no. 1, pp. 42–47, Jan. 2018.

[38] J. V. H. der, M. T. Vega, S. Petrangeli, T. Wauters, and F. de Turck, "Tile-based adaptive streaming for virtual reality video," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 15, no. 4, pp. 1–24, Dec. 2019.

[39] C. C. Chi, M. Alvarez-Mesa, B. Juurlink, G. Clare, F. Henry, S. Pateux, and T. Schierl, "Parallel scalability and efficiency of hevc parallelization approaches," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1827–1838, Dec. 2012.

[40] T. Amestoy, W. Hamidouche, C. Bergeron, and D. Menard, "Quality-driven dynamic VVC frame partitioning for efficient parallel processing," Oct. 2020, pp. 3129–3133.

[41] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, "Tile adaptation for workload balancing of 3D-HEVC encoder in homogeneous multicore systems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 5, pp. 1704–1714, Mar. 2020.

[42] T. Li, L. Yu, H. Wang, and Z. Kuang, "A bit allocation method based on inter-view dependency and spatio-temporal correlation for multi-view texture video coding," *IEEE Transactions on Broadcasting*, vol. 67, no. 1, pp. 159–173, Mar. 2021.

[43] M. Hegazy, K. M. Diab, M. Saeedi, B. Ivanovic, I. Amer, Y. Liu, G. Sines, and M. Hefeeda, "Content-aware video encoding for cloud gaming," in *Proceedings of the 10th ACM Multimedia Systems Conference*, ser. MMSys '19, Jun. 2019, pp. 60–73.

[44] S. S. Sabet, M. R. Hashemi, S. Shirmohammadi, and M. Ghanbari, "A novel objective quality assessment method for perceptually-coded cloud gaming video," in *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, Apr. 2018, pp. 75–79.

[45] G. K. Illahi, T. V. Gemert, M. Siekkinen, E. Masala, A. Oulasvirta, and A. Ylä-Jääski, "Cloud gaming with foveated video encoding," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 16, no. 1, pp. 1–24, Feb. 2020.

[46] C. C. Chi, M. Alvarez-Mesa, B. Juurlink, G. Clare, F. Henry, S. Pateux, and T. Schierl, "Parallel scalability and efficiency of HEVC parallelization approaches," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1827–1838, Dec. 2012.

[47] D.-Y. Chen and M. El-Zarki, "A framework for adaptive residual streaming for single-player cloud gaming," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 15, no. 2s, Apr. 2019.

[48] Y. Han, D. Guo, W. Cai, X. Wang, and V. C. M. Leung, "Virtual machine placement optimization in mobile cloud gaming through QoE-Oriented resource competition," *IEEE Transactions on Cloud Computing*, vol. 10, no. 3, pp. 2204–2218, Jun. 2022.

[49] W. Cai, R. Shea, C.-Y. Huang, K.-T. Chen, J. Liu, V. C. M. Leung, and C.-H. Hsu, "A survey on cloud gaming: Future of computer games," *IEEE Access*, vol. 4, pp. 7605–7620, Aug. 2016.

[50] Y. Deng, Y. Li, R. Seet, X. Tang, and W. Cai, "The server allocation problem for session-based multiplayer cloud gaming," *IEEE Transactions on Multimedia*, vol. 20, no. 5, pp. 1233–1245, May 2018.

[51] Y. Li, Y. Deng, X. Tang, W. Cai, X. Liu, and G. Wang, "Cost-efficient server provisioning for cloud gaming," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 14, no. 3s, Jun. 2018.

[52] M. Luo and M. Claypool, "Uniquitous: Implementation and evaluation of a cloud-based game system in unity," in *2015 IEEE Games Entertainment Media Conference (GEM)*, Oct. 2015, pp. 1–6.

[53] W. Cai, Y. Chi, C. Zhou, C. Zhu, and V. C. M. Leung, "UBCGaming: Ubiquitous cloud gaming system," *IEEE Systems Journal*, vol. 12, no. 3, pp. 2483–2494, Sep. 2018.

[54] M. H. Yami, F. Pakdaman, and M. R. Hashemi, "SARA-SDN: State aware resource allocation in SDN to improve QoE in cloud gaming," in *Proceedings of the 25th ACM Workshop on Packet Video*, ser. PV '20. Association for Computing Machinery, Jun. 2020, pp. 8–14.

[55] M. Amiri, H. A. Osman, and S. Shirmohammadi, "Resource optimization through hierarchical SDN-Enabled inter data center network for cloud gaming," in *Proceedings of the 11th ACM Multimedia Systems Conference*, ser. MMSys '20. Association for Computing Machinery, 2020, pp. 166–177.

[56] M. Carrascosa and B. Bellalta, "Cloud-gaming: Analysis of google stadia traffic," *Computer Communications*, vol. 188, pp. 99–116, Apr. 2022.

[57] A. D. Domenico, G. Perna, M. Trevisan, L. Vassio, and D. Giordano, "A network analysis on cloud gaming: Stadia, GeForce now and PSNow," *Network*, vol. 1, no. 3, pp. 247–260, 2021.

[58] A. Alós, F. Morán, P. Carballeira, D. Berjń, and N. García, "Congestion control for cloud gaming over udp based on Round-Trip video latency," *IEEE Access*, vol. 7, pp. 78 882–78 897, Jun. 2019.

[59] J. Wu, C. Yuen, N.-M. Cheung, J. Chen, and C. W. Chen, "Streaming mobile cloud gaming video over TCP with adaptive source-FEC coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 1, pp. 32–48, 2017.

[60] A. Faisal and M. Zulkernine, "A secure architecture for TCP/UDP-based cloud communications," *International Journal of Information Security*, vol. 20, no. 2, pp. 161–179, Apr. 2021.

[61] X. Liao, L. Lin, G. Tan, H. Jin, X. Yang, W. Zhang, and B. Li, "LiveRender: A cloud gaming system based on compressed graphics streaming," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2128–2139, Aug. 2016.

[62] Y. Li, C. Zhao, X. Tang, W. Cai, X. Liu, G. Wang, and X. Gong, "Towards minimizing resource usage with QoS guarantee in cloud gaming," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 2, pp. 426–440, Feb. 2021.

[63] H. Nammas and M. Quwaider, "Yet efficient study for evaluating the quality of service of cloud gaming systems," in *2022 13th International Conference on Information and Communication Systems (ICICS)*, Jun. 2022, pp. 1–6.

[64] S. Flinck Lindström, M. Wetterberg, and N. Carlsson, "Cloud gaming: A QoE study of fast-paced single-player and multiplayer gaming," in *2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC)*, Dec. 2020, pp. 34–45.

[65] A. A. Laghari, H. He, A. Khan, N. Kumar, and R. Kharel, "Quality of experience framework for cloud computing (QoC)," *IEEE Access*, vol. 6, pp. 64 876–64 890, Aug. 2018.

[66] P. Graff, X. Marchal, T. Cholez, B. Mathieu, and O. Festor, "Efficient identification of cloud gaming traffic at the edge," in *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*, May 2023, pp. 1–10.

[67] B. Baldovino, "An overview of the networking issues of cloud gaming: A literature review," *Journal of Innovation Information Technology and Application (JINITA)*, vol. 4, pp. 120–132, Dec. 2022.

[68] Y.-T. Lee, K.-T. Chen, H.-I. Su, and C.-L. Lei, "Are all games equally cloud-gaming-friendly? an electromyographic approach," in *2012 11th Annual Workshop on Network and Systems Support for Games (NetGames)*, Nov. 2012, pp. 1–6.

[69] R. D. Yates, M. Tavan, Y. Hu, and D. Raychaudhurir, "Timely cloud gaming," in *IEEE INFOCOM 2017 IEEE Conference on Computer Communications*, May 2017.

[70] S. S. Sabet, S. Schmidt, S. Zadtootaghaj, B. Naderi, C. Griwodz, and S. Möller, "A latency compensation technique based on game characteristics to mitigate the influence of delay on cloud gaming quality of experience," in *Proceedings of the 11th ACM Multimedia Systems Conference*, May 2020, pp. 15–25.

[71] R. Salay and M. L. Claypool, "A comparison of automatic versus manual world alteration for network game latency compensation," in *Extended Abstracts of the 2020 Annual Symposium on Computer-Human Interaction in Play*, Nov. 2020, pp. 355–359.

[72] J. Kim, P. Knowles, J. Spjut, B. Boudaoud, and M. Mcguire, "Post-render warp with late input sampling improves aiming under high latency conditions," vol. 3, no. 2, Aug. 2020.

[73] S. S. Sabet, S. Schmidt, S. Zadtootaghaj, C. Griwodz, and S. Moller, "Towards the impact of gamers strategy and user inputs on the delay sensitivity of cloud games," in *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, May 2020, pp. 1–3.

[74] J. Bulman and P. Garraghan, "A cloud gaming framework for dynamic graphical rendering towards achieving distributed game engines," in *Proceedings of the 12th USENIX Conference on Hot Topics in Cloud Computing*, ser. HotCloud '20. USENIX Association, 2020.

[75] H. Iqbal, A. Khalid, and M. Shahzad, "Dissecting cloud gaming performance with DECAF," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 5, no. 3, Dec. 2021.

[76] L. D. Giovanni, D. Gadia, P. Giaccone, D. Maggiorini, C. E. Palazzi, L. A. Ripamonti, and G. Sviridov, "Revamping cloud gaming with distributed engines," *IEEE Internet Computing*, vol. 26, no. 6, pp. 88–95, May 2022.

[77] X. Deng, J. Zhang, H. Zhang, and P. Jiang, "Deep-Reinforcement-Learning-Based resource allocation for cloud gaming via edge computing," *IEEE Internet of Things Journal*, vol. 10, no. 6, pp. 5364–5377, Nov. 2022.

[78] N. Justesen, P. Bontrager, J. Togelius, and S. Risi, "Deep learning for video game playing," *IEEE Transactions on Games*, vol. 12, no. 1, pp. 1–20, Feb. 2020.

[79] E. A. Romero-Mendez, P. C. Santana-Mancilla, M. Garcia-Ruiz, O. A. Montesinos-López, and L. E. Anido-Rifón, "The use of deep learning to improve player engagement in a video game through a dynamic difficulty adjustment based on skills classification," *Applied Sciences*, vol. 13, no. 14, Jul. 2023.

[80] "GGPO rollback networking SDK," https://www.ggpo.net/, accessed: 19 May 2023.

[81] A. Ehlert, "Improving input prediction in online fighting games," Master's thesis, 2021.

[82] M. Zamith, J. R. da Silva Junior, E. W. G. Clua, and M. Joselli, "Applying hidden markov model for dynamic game balancing," in *2020 19th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, Nov. 2020, pp. 38–46.

[83] G. L. Zuin and Y. P. A. Macedo, "Attempting to discover infinite combos in fighting games using hidden markov models," in *2015 14th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, Nov. 2015, pp. 80–88.

[84] K. Yu and N. R. Sturtevant, "Application of retrograde analysis on fighting games," in *2019 IEEE Conference on Games (CoG)*, Sep. 2019, pp. 1–8.

[85] K. Asayama, K. Moriyama, K. ichi Fukui, and M. Numao, "Prediction as faster perception in a real-time fighting video game," in *2015 IEEE Conference on Computational Intelligence and Games (CIG)*, Aug. 2015, pp. 517–522.

[86] N. Justesen and S. Risi, "Learning macromanagement in starcraft from replays using deep learning," in *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, Aug. 2017, pp. 162–169.

[87] H.-C. Cho, K.-J. Kim, and S.-B. Cho, "Replay-based strategy prediction and build order adaptation for starcraft ai bots," in *2013 IEEE Conference on Computational Inteligence in Games (CIG)*, Aug. 2013, pp. 1–7.

[88] B. Anand and P. Wenren, "CloudHide: Towards latency hiding techniques for thin-client cloud gaming," in *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, ser. Thematic Workshops '17, Oct. 2017, pp. 144–152.

[89] K. Lee, D. Chu, E. Cuervo, J. Kopf, S. Grizan, A. Wolman, and J. Flinn, "Outatime: Using speculation to enable Low-Latency continuous interaction for mobile cloud gaming," *MobiSys 2015 - Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, pp. 151–165, Jun. 2015.

[90] M. T. Sadaike and G. Bressan, "Optimization of quality of service for a cloud gaming system using layer catching and motion prediction," in *Proceedings of the 22nd Brazilian Symposium on Multimedia and the Web*, ser. Webmedia '16. Association for Computing Machinery, Nov. 2016, pp. 219–222.

[91] B. Boudaoud, P. Knowles, J. Kim, and J. Spjut, "Gaming at warp speed: Improving aiming with late warp," in *ACM SIGGRAPH 2021 Emerging Technologies*, ser. SIGGRAPH '21, Aug. 2021.

[92] S. Zadtootaghaj, S. Schmidt, and S. Möller, "Modeling gaming QoE: Towards the impact of frame rate and bit rate on cloud gaming," pp. 1–6, May 2018.

[93] "G.1072 : Opinion model predicting gaming quality of experience for cloud gaming services," https://www.itu.int/rec/T-REC-G.1072/en, accessed: 1 September 2021.

[94] S. Schmidt, S. Zadtootaghaj, and S. Möller, "Towards the delay sensitivity of games: There is more than genres," in *2017 Ninth International Conference on Quality of Multimedia Experience (QoMEX)*, May 2017, pp. 1–6.

[95] C. Zauner, "Implementation and benchmarking of perceptual image hash functions," 2010.

[96] W. Hua, M. Hou, Y. Qiao, X. Zhao, S. Xu, and S. Li, "Similarity index based approach for identifying similar grotto statues to support virtual restoration," *Remote Sensing*, vol. 13, no. 6, 2021.

[97] "VALORANT official webcite," https://playvalorant.com/, accessed: 28 June 2022.

[98] "Overwatch official webcite," https://playoverwatch.com/, accessed: 28 June 2022.

[99] "Borderlands official webcite," https://borderlands.com/, accessed: 28 June 2022.

[100] "GenshinImpact official webcite," https://genshin.hoyoverse.com/, accessed: 28 June 2022.

[101] "MONSTER HUNTER WORLD official webcite," https://www.monsterhunterworld.com/, accessed: 28 June 2022.

[102] "ELDENRING official webcite," https://www.eldenring.com/, accessed: 28 June 2022.

[103] "STREET FIGHTER V CHAMPION EDITION official webcite," https://www.capcom.co.jp/sfv/, accessed: 28 June 2022.

[104] "Hearthstone official webcite," https://playhearthstone.com/, accessed: 28 June 2022.

[105] "Cuphead official webcite," https://cupheadgame.com/, accessed: 28 June 2022.

[106] X. Foukas, N. Nikaein, M. Kassem, M. K. Marina, and K. Kontovasilis, "FlexRAN: A flexible and programmable platform for software-defined radio access networks," in *Proceedings of the 12th International on Conference on Emerging Networking Experiments and Technologies (CoNEXT'16)*, Nov. 2016, pp. 427–441.

[107] I. F. Akyildiz, P. Wang, and S.-C. Lin, "SoftAir: A software defined networking architecture for 5G wireless systems," *Computer Networks*, vol. 85, May 2015.

[108] M. Peng, Y. Li, J. Jiang, J. Li, and C. Wang, "Heterogeneous cloud radio access networks: A new perspective for enhancing spectral and energy efficiencies," *IEEE Wireless Communications*, vol. 21, no. 6, pp. 126–135, Dec. 2014.

[109] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog-computing-based radio access networks: issues and challenges," *IEEE Network*, vol. 30, no. 4, pp. 46–53, Jul. 2016.

[110] J. H. Cox, J. Chung, S. Donovan, J. Ivey, R. J. Clark, G. Riley, and H. L. Owen, "Advancing software-defined networks: A survey," *IEEE Access*, vol. 5, pp. 25 487–25 526, October 2017.

[111] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca, and J. Folgueira, "Network slicing for 5G with SDN/NFV: Concepts, architectures and challenges," *IEEE Communications Magazine*, vol. 55, pp. 80–87, May 2017.

[112] K. A. Noghani, C. H. Benet, A. Kassler, A. Marotta, P. Jestin, and V. V. Srivastava, "Automating ethernet VPN deployment in SDN-based data centers," in *International Conference on Software Defined Systems (SDS'17)*, May 2017.

[113] D. K. Hong, Y. Ma, S. Banerjee, and Z. M. Mao, "Incremental deployment of SDN in hybrid enterprise and ISP networks," in *Proceedings of the Symposium on SDN Research*. Association for Computing Machinery, Mar. 2016, pp. 1–7.

[114] J. Yu, X. Li, and W. Zhou, "Tutorial: Broadband fiber-wireless integration for 5g+ communication," *APL Photonics*, vol. 3, no. 11, p. 111101, 2018.

[115] S. Deronne, V. Moeyaert, and S. Bette, "Analysis of the MAC performances in 802.11g radio-over-fiber systems," in *2011 18th IEEE Symposium on Communications and Vehicular Technology in the Benelux (SCVT)*, November 2011, pp. 1–5.

[116] K. Funabiki, T. Nishio, M. Morikura, K. Yamamoto, D. Murayama, and K. Nakahira, "ATRAS: adaptive MAC protocol for efficient and fair coexistence between radio over fiber-based and CSMA/CA-based WLANs," *Eurasip Journal on Wireless Communications and Networking*, vol. 2017, no. 1, July 2017.

[117] Y. Katsuyama, T. Sato, Z. Wen, X. Qi, K. Tamesue, W. Kameyama, Y. Nakamura, T. Sato, and J. Katto, "A predictive approach for compensating transmission latency in remote robot control for improving teleoperation efficiency," in *2023 IEEE Global Communications Conference (GLOBECOM): Communications Software and Multimedia*, Dec. 2023, pp. 1–6.