IEICE TRANSACTIONS

on Communications

DOI:10.23919/transcom.2023EBP3108

This advance publication article will be replaced by the finalized version after proofreading.

A PUBLICATION OF THE COMMUNICATIONS SOCIETY

The Institute of Electronics, Information and Communication Engineers

Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER

Traffic Reduction for Speculative Video Transmission in Cloud Gaming Systems

Takumasa ISHIOKA[†], **Tatsuya FUKUI**[††], **Toshihito FUJIWARA**[††], **Satoshi NARIKAWA**[††], **Takuya FUJIHASHI**[†], **Shunsuke SARUWATARI**[†], ***and*** **Takashi WATANABE**[†],

SUMMARY Cloud gaming systems allow users to play games that require high-performance computational capability on their mobile devices at any location. However, playing games through cloud gaming systems increases the Round-Trip Time (RTT) due to increased network delay. To simulate a local gaming experience for cloud users, we must minimize RTTs, which include network delays. The speculative video transmission pre-generates and encodes video frames corresponding to all possible user inputs and sends them to the user before the user's input. The speculative video transmission mitigates the network, whereas a simple solution significantly increases the video traffic. This paper proposes tile-wise delta detection for traffic reduction of speculative video transmission. More specifically, the proposed method determines a reference video frame from the generated video frames and divides the reference video frame into multiple tiles. We calculate the similarity between each tile of the reference video frame and other video frames based on a hash function. Based on calculated similarity, we determine redundant tiles and do not transmit them to reduce traffic volume in minimal processing time without implementing a high compression ratio video compression technique. Evaluations using commercial games showed that the proposed method reduced 40–50% in traffic volume when the SSIM index was around 0.98 in certain genres, compared with the speculative video transmission method. Furthermore, to evaluate the feasibility of the proposed method, we investigated the effectiveness of network delay reduction with existing computational capability and the requirements in the future. As a result, we found that the proposed scheme may mitigate network delay by one to two frames, even with existing computational capability under limited conditions.

key words: *cloud gaming, low latency, speculative video transmission, traffic reduction*

1. Introduction

As networks become more sophisticated, there is a growing interest in cloud gaming services [1]. A cloud gaming service enables high-end games on low-end devices by running games on a cloud server equipped with a GPU to replace the graphics rendering process [2]. High-end games require complex processing with dedicated GPU cards installed in high-performance PCs or game consoles. Using cloud gaming services, users can play high-end games from anywhere with mobile terminals like smartphones.

Cloud gaming systems necessitate interactive operation over the network. Typical video transmission for cloud gaming services is request-and-response. Specifically, a user device sends input to the cloud server, and the server sends back video frames corresponding to the received input. This generated video frame is subsequently compressed using video coding technology and transmitted to the user. The user decodes the received video frame and displays the video frame. However, a significant problem with the cloud gaming system is an increase in the round-trip time (RTT). RTT is the delay from sending the user's input to displaying the corresponding video frame on the user's device. The cloud gaming server must wait for the input to return the video frames, increasing the network delay. The network delay is the sum of the packet delivery time from the user's device to the server and vice-versa, and it is a part of the RTT. Here, the packet delivery time represents the delay when the first bit leaves the sender until the last is received. Any increase in the RTT negatively impacts the quality of the user experience, thus posing a major challenge in gameplay [3].

Many discussions have addressed the issue of RTT on cloud gaming systems. The RTT has a significant effect on the gaming experience [4]. It's a crucial factor in some games, and the demands can change based on the target game's features [5]. Beyond reducing propagation delay, several aspects, like reducing transmission delay through video quality optimization, application optimization, and more, have been explored to cut down on system-wide latency. Table 1 lists major research topics related to cloud gaming response time and traffic issues. Certain studies have used platform techniques, like mobile edge computing, to shrink response time [6]. Numerous studies focused on platform optimization, which includes dynamically adjusting video quality [7], optimizing cloud resources [8, 9], and creating cloud-native games [10, 11]. However, simply reducing latency does not provide an experience identical to playing a game locally. We need to eliminate any network delays to make the user's experience match that of playing a game locally.

Some studies suggest speculative video transmission methods to minimize the RTT. Speculative video transmission is an approach where video data is transmitted efficiently by utilizing input predictions. This approach predicts the next potential input based on the user's past inputs or general input patterns and accordingly pre-renders and transmits the frames. For example, in [12], the cloud server creates in advance and encodes video frames corresponding to all potential user inputs within one frame, then sends them all to the user. The speculative video transmission aims to mitigate the network delay. In other words, we aim for the perceived network delay to be zero. The perceived network

[†]Graduate School of Information Science and Technology, Osaka University, Osaka 565-0871, Japan

[††]NTT Access Network Service Systems Laboratories, NTT Corporation, 3-9-11 Midori-cho, Musashino-shi, Tokyo, 180-8585, Japan

Table 1: Major research topics on response time and traffic challenges in cloud gaming

Research Topics	**Description**
Low-latency Network Techniques	Developing and applying high-speed, low-latency network technologies (5G/6G) to mitigate latency and improve the cloud gaming experience.
Edge Computing	Processing games on devices closer to the edge of the network to mitigate response time and traffic issues.
Dynamic Video Quality Adjustment	Dynamically adjusting video quality according to bandwidth constraints to mitigate response time and traffic issues.
Data Compression Techniques	Developing and applying efficient compression techniques for images and audio to reduce data transfer volume, thereby mitigating response time and traffic issues.
Cloud Resource Allocation Optimization	Optimizing cloud resource allocation and developing algorithms and methods to mitigate response time and traffic issues.
Network Protocol Optimization	Developing and applying new network protocols and techniques to mitigate response time and traffic issues.
Cloud-native Game Development	Developing games optimized for the cloud environment to address response and traffic issues and improve the cloud gaming experience.
Latency Mitigation using Deep Reinforcement Learning	Optimizing cloud gaming experience using deep reinforcement learning to mitigate response time issues.
Speculative Execution on Cloud Gaming Servers **(A topic of focus in this paper)**	Pre-rendering game frames corresponding to all potential user input patterns and transmitting the frames in advance to mitigate response time issues, ideally achieving zero latency.

delay is essentially how a user experiences the network delay. By transmitting video frames speculatively, the proposed scheme makes the user perceive the network delay as zero, even if there's an inherent delay.

A key issue in realizing speculative video transmission is significant video traffic when the number of the user's potential inputs in each frame is large. For example, when the network delay is d [s] and the frame rate is f [fps], the server lists the potential inputs in future $n = \lceil d \cdot f \rceil$ frames. The server then renders the video frames according to the potential inputs and transmits the video frames to the user. Since the video frames corresponding to the listed potential inputs are buffered on the user's device before his/her input, the network delay can be regarded as zero. The server then renders the video frames according to the listed potential inputs and transmits the video frames to the user. Current request-and-response transmissions commonly use differential coding methods such as H.264/Advanced Video Coding (AVC) [13] and H.265/High Efficiency Video Coding (HEVC) [14]. However, each speculative video frame contains the effects of prediction errors and uncertainty in prediction data. It is desirable to mitigate the impact of accumulated prediction errors and the uncertainty of prediction data by processing each frame independently. Therefore, in speculative video transmission, intra-coding, a coding method that processes each frame independently, is a more appropriate approach than the existing differential coding methods. If one simply uses intra-coding alone, it would be challenging to achieve sufficient compression rates to counteract the increase in traffic volume caused by speculative processing. The proposed method aims to reduce traffic volume by focusing on the similarities between speculative frames at the same time slot, utilizing intra-coding as a base.

As a method focusing on the similarities between speculative frames within the same time slot, this paper proposes a tile-wise delta detection approach. In this approach, the cloud server generates the video frames based on the user's potential input and selects one video frame as the reference frame. It then partitions these generated video frames into multiple tiles. Instead of encoding and transmitting all tiles uniformly, the system sends only those that have notable differences from the tiles of the reference frame. Furthermore, this tile-wise delta detection approach utilizes a hash function to estimate the similarity between the tiles of the reference frame and the generated videos. This approach achieves a low computational cost while maintaining certain similarity estimation accuracy.

To evaluate the effectiveness of the proposed method, we adopted several commercial games, including first-person, third-person, and omniscient games, for comparison. From the evaluations, the proposed method reduces the video traffic for speculative video transmission irrespective of the game genre. Depending on the game genres, we achieve high reductions especially. In addition, we discuss the feasibility of the proposed method considering the computational capability of the cloud server.

The contributions of this study are as follows:

1. We design tile-wise delta detection for cloud gaming services to simultaneously achieve network delay mitigation and traffic reduction while keeping video quality.
2. We adopt a hash-based similarity calculation to estimate the similarity between the tiles with a low computational cost.
3. We develop a speculative video transmission system to empirically evaluate the effectiveness and feasibility of the proposed method.
4. We use three genres of commercial games for evaluations, i.e., first-person, third-person, and omniscient games, to further discuss the effectiveness of the proposed method in practical game videos.

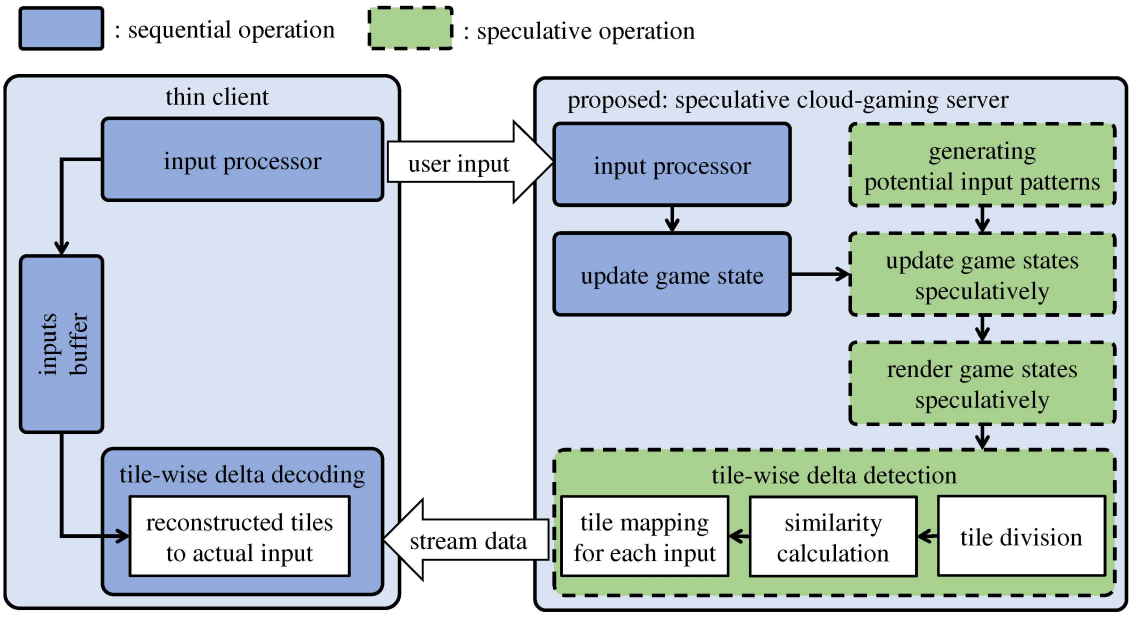


Fig. 1: Proposed cloud gaming system architecture

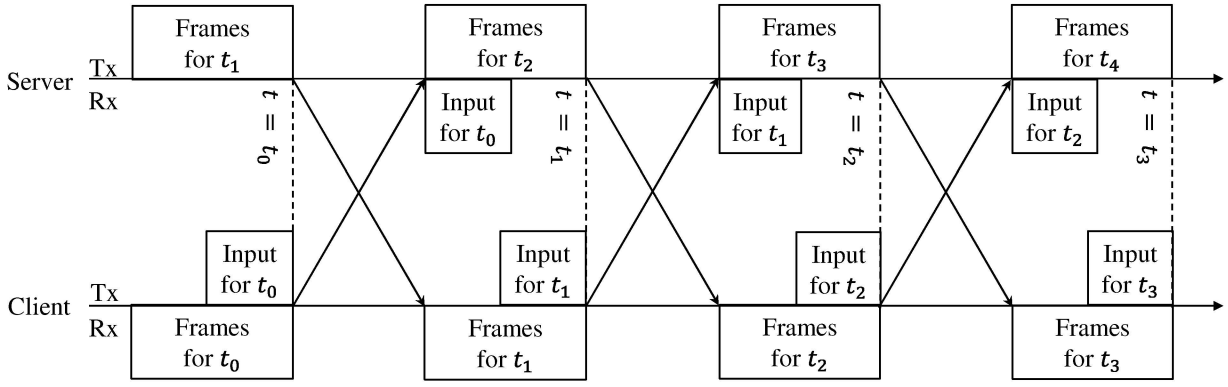


Fig. 2: Data flow between server and client

5. We simulated the computational capability required to implement the proposed method and discussed the feasibility.

2. Proposed Method

2.1 Overview

Fig. 1 shows an overview architecture of the proposed method. The cloud gaming server generates all potential input patterns for the user according to the number of speculative frames and input patterns per frame. The system renders the video frames according to the potential input patterns and the current game state. It divides each rendered video frame into multiple tiles and selects one video frame as the reference to calculate the similarity between other video frames using a hash function. It calculates the similarity for each tile. Based on the similarity, the tiles with high similarity are not encoded and transmitted to the user. The user uses the same tiles in the reference video frame, whereas the tiles with low similarity are encoded and transmitted to the user. Each user decodes the video frame corresponding to the actual input and displays it on the screen. The cloud gaming server begins the transmission of the video frames, considering the network delay. The network delay appears to be zero from the user's perspective. In addition, each user sequentially sends his/her input to the cloud gaming server to update the current game state.

Fig. 2 shows the data flow between the cloud gaming server and the user's device. We consider the number of speculative frames n to be two based on the network delay and the frame rate. At the time instance t_0, the user's device sends an input to the cloud gaming server. The cloud gaming server has already received the input for t_{-1} from the user's device and updates the game state based on the received input. Let A be the number of the potential input patterns in each frame. In this case, the cloud gaming server generates A^2 potential input patterns and prepares multiple game states

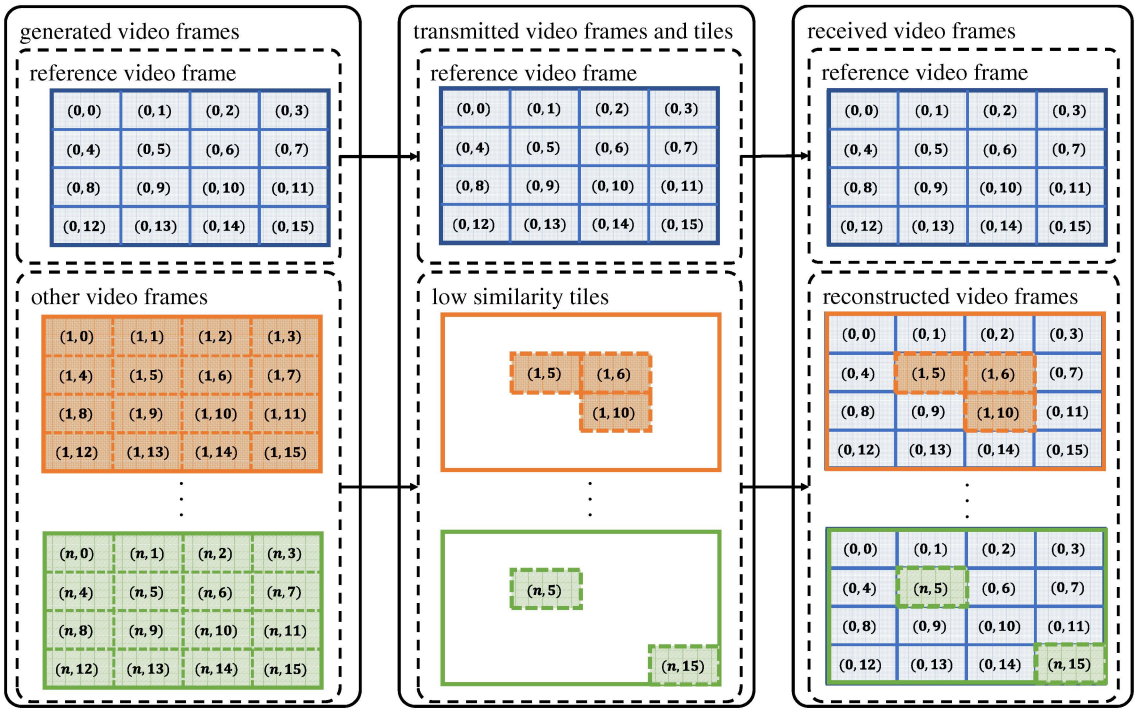


Fig. 3: Overview of the proposed tile-wise delta detection

for t_1 by speculatively updating the game state for each input. It then renders video frames of the game states for t_1 and sends the video frames to the user's device, followed by the proposed tile-wise delta detection. The user's device displays the video frames with a negligible RTT based on the user's true input at t_1. The above operations are repeated in every frame.

2.2 Tile-wise Delta Detection

Fig. 3 shows the overview of the proposed tile-wise delta detection. The cloud gaming server with speculative video transmission generates video frames for user potential input patterns. The speculative video transmission can mitigate the network delay by pre-sending the generated video frames to the user, whereas it will cause large video traffic. In the proposed method, we detect redundant frames with respect to the reference frame. It then calculates the similarity between the reference video frame and the other tiles. For the similarity calculation, the cloud gaming server partitions the reference video and other video frames into M vertical tiles and N horizontal tiles. Fig. 3 presents an example for the case where $M = 4$ and $N = 4$. We identify redundant tiles relative to the reference frame to reduce the video traffic generated by video frames.

Our proposed method employs perceptual hash (pHash) [15] to calculate the similarity based on Discrete Cosine Transform (DCT) between tiles. The pHash is a widely utilized image similarity estimation algorithm. It prioritizes extracting the features in the low-frequency components easily perceived by humans, thereby enabling the extraction of perceptible differences. Despite requiring longer computational time compared to average hash (aHash) and difference hash (dHash), pHash demonstrates robustness against image reduction and blur [16]. Considering that certain games may generate high-quality images or motion blur, we argue that pHash is our method's most appropriate image similarity estimation algorithm. The procedure to compute the pHash value for each tile is as follows.

1. Converts a video frame to a grayscale image with the same luminance as the original color image.
2. Resize the grayscale image to 32×32 [pixels].
3. Perform a discrete cosine transform (DCT) for the resized tiles and extract the DCT coefficients over an 8×8 [pixels] region. It selects 64 low-frequency DCT coefficients.
4. Obtain the median of the extracted DCT coefficients and convert them to binary based on the average value. Finally, we obtain a 64-bit hash value.

The system then uses the pHash values corresponding to each tile in the reference video frame and the hash values of tiles in other video frames to determine similar and dissimilar tiles according to Hamming distance. The maximum possible Hamming distance is 64.

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:

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

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

$$P_S = P_G \cdot A^{\hat{n}} \tag{2}$$

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

$$\hat{n} = \left\lfloor \frac{1}{2} \log_A \left(\frac{C_S}{f \cdot P_G} \right) \right\rfloor \tag{3}$$

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:

$$t_p = \begin{cases} 0 & \text{if } \left(\frac{\hat{n}}{f} \geq d \right), \\ d - \frac{\hat{n}}{f} & \text{else} \end{cases} \tag{4}$$

The proposed method reduces the perceived network delay by rendering a large number of future input patterns frame-by-frame. However, as defined in Eq. (3), significant computation power is required to mitigate the perceived network delay as the value of $\hat{n}$ increases.

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 [17] 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 not always 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:

$$\mathcal{F}_t = \{F_k | k \leq A^{\hat{n}}\} \tag{5}$$

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:

$$D = \sum_{k=1}^{A^{\hat{n}}} D(r(k)) P(F_k|\mathcal{F}) \tag{6}$$

In the proposed method, the difference from the reference frame is determined for each tile of M×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. It means that setting a lower hash threshold increases the number of transmission tiles. In this case, the distortion is defined as:

$$\begin{aligned} D &= \sum_{k=1}^{A^{\hat{n}}} \sum_{i=1}^{\mathrm{M}} \sum_{j=1}^{\mathrm{N}} D(r(i, j, k)) P(F_k|\mathcal{F}) x(i, j, k) \\ &s.t. \quad \sum_{i=1}^{\mathrm{M}} \sum_{j=1}^{\mathrm{N}} \sum_{k=1}^{A^{\hat{n}}} r(i, j, k) x(i, j, k) \leq R_{limit} \\ &s.t. \quad x(i, j, k) = \begin{cases} 1 & \text{(if send)} \\ 0 & \text{(else)} \end{cases} \end{aligned} \tag{7}$$

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

$$\begin{aligned} J = \min_{\mathbf{r},\mathbf{x}} \Bigg\{ &\sum_{k=1}^{A^{\hat{n}}} \sum_{i=1}^{\mathrm{M}} \sum_{j=1}^{\mathrm{N}} D(r(i, j, k)) P(F_k|\mathcal{F}) x(i, j, k) \\ &+ \lambda ||\mathbf{rx}||_1 \Bigg\} \end{aligned} \tag{8}$$

Table 2: PC specs used for evaluation

	configuration
OS	Ubuntu Desktop 20.04.2.0 LTS
Motherboard	Z490 1200 DDR4 ATX
CPU	Intel Core i9-10850K
RAM-size	128 GB
GPU	NVIDIA GeForce RTX 3080

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. Evaluation

4.1 Traffic Reduction in Commercial Games

We measured the video traffic and the corresponding structural similarity (SSIM) index [18] of the speculative video transmission with tile-wise delta detection. Table 2 presents the environments of this evaluation. SSIM index is a better objective metric for predicting the perceptual similarity between original and processed video frames. Larger values of the SSIM index close to 1 indicate higher perceptual similarity between these frames.

In this evaluation, we classify commercial games into first-person, third-person, and omniscient. First-person games are displayed from the viewpoint of the user-controlled character. First-person shooting (FPS) is a typical first-person game. The video frame of an FPS game is characterized by the display of the user interface (UI) and weapons, such as guns and knives held by the characters at certain 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 certain 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 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 [19], Overwatch2 (OW2) [20], and Borderlands3 [21] as first-person games; Genshin Impact [22], Monster Hunter World (MHW) [23], and ELDENRING [24] as third-person games; and Street Fighter V (SFV) [25], Hearthstone [26], and Cuphead [27] 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 to serve as the reference frame. All frames, also divided tiles, were processed without compression. Accordingly, in Equation (8), $P(F_k|\mathcal{F})$ remains constant regardless of the values of k, and $D(\mathbf{r}(i, j, k))$ equals zero.

Table 3 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(a) through 4(f) show the snapshots of the games. We select Valorant, Genshin Impact, and SFV as the first-person, third-person, and omniscient games. Figures 4(a) through 4(c) show the original video frames and Figures 4(d) through 4(f) 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 5(a) through 5(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 5(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

Table 3: 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

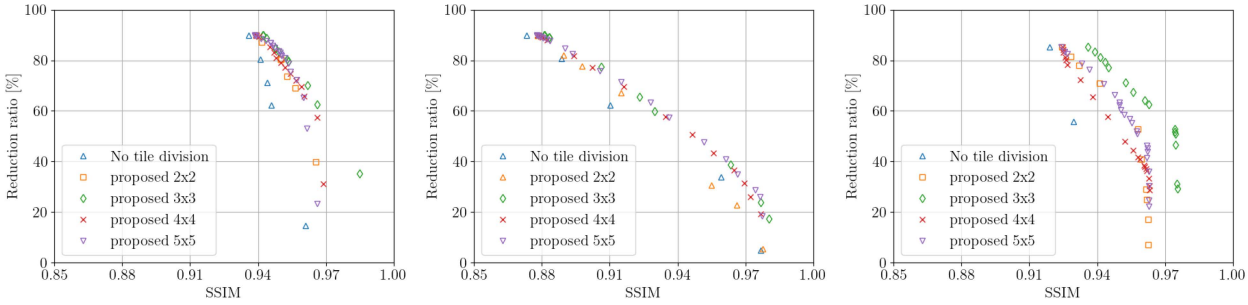
(a) Original (Valorant) (b) Original (Genshin Impact) (c) Original (SFV)

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

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

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

Fig. 4: Snapshots of the games. (a) and (d) show the first-person game's video frames. (b) and (e) shows the third-person game's video frames. (c) and (f) shows the omniscient game's video frames.



(a) first-person game (Valorant) (b) third-person game (Genshin Impact) (c) omniscient game (SFV)

Fig. 5: 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.

tiles.

- Fig. 5(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.
- Fig. 5(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, and this can lead to a large reduction in traffic.
- Fig. 5(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 3, the results from Figures 5(a) to 5(c) suggest that

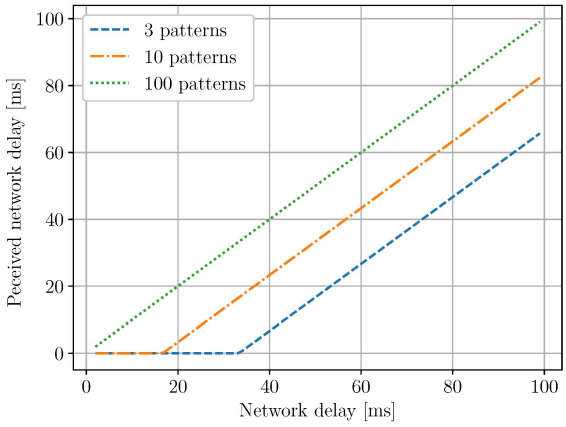


Fig. 6: The perceived network delay as a function of network delays with the different number of potential input patterns in each frame.

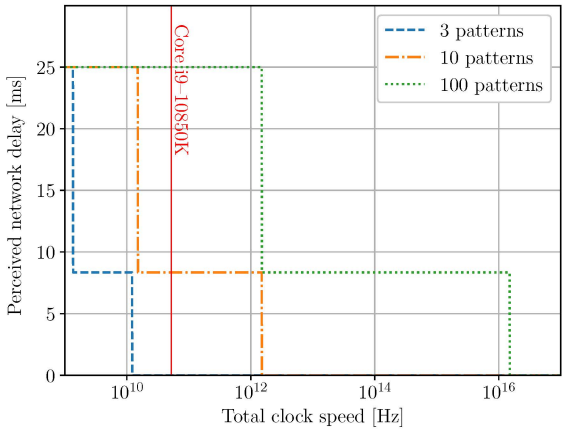


Fig. 7: The perceived network delay as a function of the total clock speed, where the network delay is assumed to be 25 ms.

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.

4.2 Feasibility

In this section, we carry out experiments to discuss the perceived network delay under the different computation resources and the number of potential input patterns in each frame. Table 2 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.

Fig. 6 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 a large number 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, also contributes to the improvement of the cloud gaming experience.

Fig. 7 shows the perceived network delay as a function of the total clock speed, where the network delay is assumed to be 25 ms [28]. 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.

5. Related Works

This research is related to studies on speculative video transmission, network delay reduction in cloud gaming, and differential coding.

5.1 Network Delay Reduction in Cloud Gaming

In cloud gaming systems, the network delay will be a critical issue since a long network delay damages the user's experience depending on the characteristics of the target game [29]. Some studies aim to reduce network delay to enhance user experience quality in cloud gaming systems. We can classify the existing methods into individual delay reduction and zero-delay solutions. For individual delay reduction, Suzujevic [30] proposed an adaptive video coding method to reduce transmission and processing delays. Specifically, the cloud gaming server degrades game video frames and their frame rate to reduce the video traffic and corresponding transmission delay. In [31], they aim to reduce propagation delay by decreasing the distance between the cloud gaming server and the user terminal. Specifically, each user connects to a physically nearby cloud gaming server and exchanges packets, thus achieving an experienced quality closer to a local gaming system. Zhang [6] utilized edge networks to reduce

network delay and bandwidth consumption. They form edge networks to a data center for cloud gaming and perform computationally demanding operations, such as video rendering, on the edge networks. Other studies [10, 11] adjust the game system to reduce the effect of a long network delay on the quality of experience.

For zero-delay solutions, speculative video transmission has been designed in recent years. Outatime [32] is proposed to eliminate the network delay in cloud gaming systems by utilizing speculative execution. Outatime transmits speculatively generated multiple video frames as early as the network delay of the server-user network. Each user outputs a game video frame from the received multiple video frames by applying the appropriate image synthesis technique according to the user's input. In contrast, the user device requires a high computation cost for running the sophisticated image synthesis technique. CloudHide [12] is another method for eliminating network delay in cloud gaming systems through speculative execution. CloudHide transmits all speculatively generated video frames to the user in advance for network delay reduction, whereas it significantly increases video traffic. In existing methods, while issues related to video traffic are mentioned, specific solutions are not provided.

Our study is one of the zero-delay solutions. We aim to eliminate the network delay by transmitting all speculatively generated game video frames to the user in advance while reducing traffic. To reduce video traffic without an additional computation cost, we propose hash function-based tile-wise delta detection. Specifically, it detects redundant tiles between game video frames using a hash function, i.e., low computation cost, and skips encoding and transmission of the redundant tiles for traffic reduction.

5.2 Differential Coding

Video coding techniques such as H.264/AVC and H.265/HEVC are used to encode video frames in cloud gaming. In H.264/AVC and H.265/HEVC, traffic reduction is achieved by encoding and transmitting the difference between the current frames and the previously encoded frame. For cloud gaming, traffic reduction methods based on users' perspectives have been proposed to prevent the degradation of the user's perceptual quality. Sabat [33] determines the video quality of the game video frames based on gazing points at objects in the game. Hegazy et al. [34] improve the perceptual quality by maximizing the video quality of the regions of interest within each video frame. Illahi [35] proposed a foveated video encoding (FVE) to determine the quality within each game video frame based on the user's gaze information. FVE can reduce traffic by encoding the periphery of the field of view at a lower resolution by taking advantage of the property that the resolution becomes lower as one moves outward from the field of view.

Other than cloud gaming, there have been studies on traffic reduction in immersive video applications. In [36], they proposed adaptive tile-based virtual reality (VR) video transmission. The tile-based video transmission schemes [37–40] divide the entire video frame into multiple tiles. There are two advantages of tile-based schemes. The first advantage is to realize video coding parallelization to decrease coding delay. For example, in [38], they realize parallel video coding in H.265/HEVC considering adaptive tile patterns and decrease the coding delay by approximately 20–40%. The second advantage is fine-grained quality control. Given that many areas in VR video are outside the user's viewport, the video quality can be improved by transmitting only the viewport. Another study in [41] aims for a traffic reduction for multi-view video. Multi-view video coding requires efficient encoding to transmit multiple viewpoint videos over band-limited networks. For this purpose, they utilize disparity prediction and compensation to remove the inter-view redundancy between the viewpoints.

The proposed tile-wise delta detection is designed for coding multiple game video frames at the same time instant. The tile-wise delta detection is a similar way to the disparity prediction, whereas the computational complexity of the tile-wise delta detection is low because of a hash function-based similarity prediction. Although the similarity prediction is a simple way, the proposed method can yield traffic reduction in commercial games from evaluation results.

6. Conclusion

In this paper, we propose a traffic reduction method for speculative video transmission in cloud gaming systems to mitigate the 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 addition, we discussed the feasibility of the proposed method based on the experimental environment. The proposed speculative execution for two frames may be possible when there are three input patterns in each frame.

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.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Numbers JP19H01101, JP22H03582, and NTT Access Network Service Systems Laboratories, Japan.

References

[1] 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.

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

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

[5] 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," 2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX), pp.1–3, May 2020.

[6] 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, April 2019.

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

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

[9] 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.

[10] 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," Proceedings of the 11th ACM Multimedia Systems Conference, pp.15–25, May 2020.

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

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

[13] 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.

[14] 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.

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

[16] 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.

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

[18] 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, April 2004.

[19] "VALORANT official webcite." URL: https://playvalorant.com/ [Accessed 28 June 2022].

[20] "Overwatch official webcite." URL: https://playoverwatch.com/ [Accessed 28 June 2022].

[21] "Borderlands official webcite." URL: https://borderlands.com/ [Accessed 28 June 2022].

[22] "GenshinImpact official webcite." URL: https://genshin.hoyoverse.com/ [Accessed 28 June 2022].

[23] "MONSTER HUNTER WORLD official webcite." URL: https://www.monsterhunterworld.com/ [Accessed 28 June 2022].

[24] "ELDENRING official webcite." URL: https://www.eldenring.com/ [Accessed 28 June 2022].

[25] "STREET FIGHTER V CHAMPION EDITION official webcite." URL: https://www.capcom.co.jp/sfv/ [Accessed 28 June 2022].

[26] "Hearthstone official webcite." URL: https://playhearthstone.com/ [Accessed 28 June 2022].

[27] "Cuphead official webcite." URL: https://cupheadgame.com/ [Accessed 28 June 2022].

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

[29] R. Shea, J. Liu, E.C.H. Ngai., and Y. Cui, "Cloud gaming: architecture and performance," IEEE Network, vol.27, no.4, pp.16–21, Aug. 2013.

[30] 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," 2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX), pp.1–6, June 2016.

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

[32] 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, June 2015.

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

[34] 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," Proceedings of the 10th ACM Multimedia Systems Conference, MMSys '19, pp.60–73, June 2019.

[35] 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.

[36] 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.

[37] 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.

[38] 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.

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

[40] 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, March 2020.
[41] 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, March 2021.

Takumasa Ishioka received the B.E. and M.E. degrees from Osaka University, Japan, in 2019 and 2021. He is currently an assistant professor at the Faculty of Engineering, Kyoto Tachibana University, Japan, since April 2023. He is a member of IEEE and IPSJ. His research interests include wireless networks.

Tatsuya Fukui received the B.E. and M.E. degrees from Waseda University, Faculty of Science and Engineering, Japan in 2008 and 2010. He is now working for NTT Access Network Service Systems Laboratories. His current research interests include research and development of carrier networks such as wide-area Ethernet systems.

Toshihito Fujiwara received the B.E., M.E., and Ph.D. degrees in engineering from the University of Tsukuba, Ibaraki, Japan, in 2002, 2004, and 2011, respectively. In 2004, he joined the NTT Access Network Service Systems Laboratories, Japan, where he has been involved in the research and development of optical video transmission system, passive optical network system, content delivery network system and ultralow latency video system.

Satoshi Narikawa received the B.E., M.E. and Ph.D. degrees from Tokyo Institute of Technology, Department of Electrical and Electronics Engineering, Japan in 2001, 2003 and 2012, respectively. He is now working for NTT Access Network Service Systems Laboratories. His current research interests include research and development of optical access systems.

Takuya Fujihashi (M'16) received the B.E. degree in 2012 and the M.S. degree in 2013 from Shizuoka University, Japan. In 2016, he received Ph.D. degree from the Graduate School of Information Science and Technology, Osaka University, Japan. He is currently an assistant professor at the Graduate School of Information Science and Technology, Osaka University since April, 2019. He was research fellow (PD) of Japan Society for the Promotion of Science in 2016. From 2014 to 2016, he was research fellow (DC1) of Japan Society for the Promotion of Science. From 2014 to 2015, he was an intern at Mitsubishi Electric Research Labs. (MERL) working with the Electronics and Communications group. His research interests are in the area of video compression and communications, with a focus on immersive video coding and streaming.

Shunsuke Saruwatari received the Dr. Sci. degree from the University of Tokyo in 2007. From 2007 to 2008, he was a visiting researcher at the IllinoisGenetic Algorithm Laboratory, University of Illinois at Urbana-Champaign. From 2008 to 2012, he was a research associate at the RCAST, the University of Tokyo. From 2012 to 2016, he was an assistant professor at Shizuoka University. He has been an associate professor at Osaka University Since 2016. His research interests are in the areas of wireless networks, sensor networks, and system software.

Takashi Watanabe (S'83–M'87) is a Professor of Graduate School of Information Science and Technology, Osaka University, Japan. He received his B.E., M.E. and Ph.D. degrees from Osaka University, Japan, in 1982, 1984 and 1987, respectively. He joined Faculty of Engineering, Tokushima University as an assistant professor in 1987 and moved to Faculty of Engineering, Shizuoka University in 1990. He was a visiting researcher at University of California, Irvine from 1995 through 1996. He has served on many program committees for networking conferences, IEEE, ACM, IPSJ, IEICE (The Institute of Electronics, Information and Communication Engineers, Japan). His research interests include mobile networking, ad hoc networks, sensornetworks, ubiquitous networks, intelligent transport systems, specially MAC and routing. He is a member of IEEE, IEEE Communications Society, IEEE Computer Society as well as IPSJ and IEICE.