Computers, Materials & Continua
DOI:10.32604/cmc.2021.013399
Article

Tech Science Press

An Efficient Viewport-Dependent 360 VR System Based on Adaptive Tiled Streaming

Tuan Thanh Le[1], Jong-Beom Jeong[2], SangSoon Lee[1], Jaehyoun Kim[2] and Eun-Seok Ryu[2,*]

[1]Department of Computer Engineering, Gachon University, Seongnam, 13120, Korea
[2]Department of Computer Education, Sungkyunkwan University, Seoul, 03063, Korea
[*]Corresponding Author: Eun-Seok Ryu. Email: esryu@skku.edu
Received: 05 August 2020; Accepted: 31 August 2020

Abstract: Recent advances in 360 video streaming technologies have enhanced the immersive experience of video streaming services. Particularly, there is immense potential for the application of 360 video encoding formats to achieve highly immersive virtual reality (VR) systems. However, 360 video streaming requires considerable bandwidth, and its performance depends on several factors. Consequently, the optimization of 360 video bitstreams according to viewport texture is crucial. Therefore, we propose an adaptive solution for VR systems using viewport-dependent tiled 360 video streaming. To increase the degrees of freedom of users, the moving picture experts group (MPEG) recently defined three degrees plus of freedom (3DoF+) and six degrees of freedom (6DoF) to support free user movement within camera-captured scenes. The proposed method supports 6DoF to allow users to move their heads freely. Herein, we propose viewport-dependent tiled 360 video streaming based on users' head movements. The proposed system generates an adaptive bitstream using tile sets that are selected according to a parameter set of user's viewport area. This extracted bitstream is then transmitted to the user's computer. After decoding, the user's viewport is generated and rendered on VR head-mounted display (HMD). Furthermore, we introduce certain approaches to reduce the motion-to-photon latency. The experimental results demonstrated that, in contrast with non-tiled streaming, the proposed method achieved high-performance 360 video streaming for VR systems, with a 25.89% BD-rate saving for Y-PSNR and 61.16% for decoding time.

Keywords: Virtual reality; 360 video; MPEG immersive; 6DoF; MCTS; viewport-dependent

1 Introduction

Currently, virtual reality (VR) technology has become widely available, and various head-mounted display (HMD) devices are available in the market. Moreover, the development of HMD devices and their ecosystems has to meet the growing demand for both service quality and experience quality. The introduction of 360 videos with up to 12K resolution, coupled with 5G high-speed connections, has significantly solved issues regarding service quality. Furthermore, users' experience of 360 videos over the internet has improved substantially, owing to the continual enhancement of connection quality. With regarding to improving the quality of experience (QoE) of VR systems, researchers have predominantly focused on two aspects: (1) user interaction and (2) the quality of service and user requirements. Moreover, various technological organizations and researchers have proposed several solutions. Certain standards, such as 3DoF, 3DoF+, and 6DoF, have been developed, typically for enhancing user experiences with regard to activities such as moving one's head, turning, and walking in a VR environment. Few studies [1,2] improved 3DoF+ and 6DoF to reduce video quality by employing various methods such as size reduction and the elimination of correlations between videos [3].

A solution for QoE enhancement, depending on the required service quality, is to optimize video transmission based on pertinent factors to reduce the bandwidth demand or delay time. Certain prospective methods are tiled-based approaches such as those elucidated in [4,5]. These methods allow the user's primary area of interest to be transmitted with high quality and the remainder of the region to be degraded to low-quality levels. The user's viewport area is extracted from the compressed bitstream using a motion-constrained tile set (MCTS) [6]. The MCTS encoder restricts inter-temporal prediction at tile boundaries and eliminates correlation between tiles. Each tile can be extracted from a bitstream and transmitted to the client. However, the existing extractor based on HEVC can only extract one tile at a time. Therefore, multiple tiled bitstreams are extracted from one original bitstream. Consequently, maintaining multiple decoders in the client is considerably challenging.

According to [7], a motion-to-photon latency of at least 20 ms is required for a 12K, 90 fps video transmission to meet the criterion of immersive quality of experience. The motion-to-photon latency is defined as the difference between the time corresponding to a user's initial movement and the time when the first image is rendered in the viewport. It includes the request time, video server processing time, time required for transmission of the extracted bitstream to the video client, decoding time, and rendering time. Therefore, reducing latency for live 360 video streams is considerably challenging.

Herein, we propose an adaptive 360 video streaming method for a single 360 video based on an MCTS tiled-based stream, as demonstrated in Fig. 1. The results of the proposed system could also be applicable to multiple 360 videos. In the proposed method, the viewport area and direction of view (FOV) are determined based on the coordinates of the viewport and movement analyses of the user's head. To this end, a video client sends a request that includes details regarding the viewport area and other metadata to a 360-video server. According to the client request information, the 360-video server calculates tiles that correspond to the user's viewport. In addition, the proposed method allows 360 video servers to extract multiple tiles from one MCTS bitstream and collate them into a single bitstream, called an adaptive tiled bitstream. Thus, the streaming server transfers the adaptive bitstream to the video client. The adaptive bitstream is then decoded and rendered to generate the user's viewport. Additionally, we developed a media delivery system based on RTSP/TCP to optimize the transmission of videos, requests, and metadata. Our method includes approaches for reducing the motion-to-photon latency and optimizing the QoE. Furthermore, our experimental results demonstrate that the proposed method could achieve viewport-dependent VR streaming with a reasonable motion-to-photon latency to feel immersive on 360 videos without motion-sickness. Finally, based on the obtained results, we can upgrade the proposed method to provide a solution for the simultaneous transmission of multiple 360 videos based on OMAF [8,9], TMIV [10], and 6DoF 360 video streaming.

The remainder of this paper is organized as follows: Section 2 elucidates the background to our study and provides a brief overview of the literature on VR streaming. Section 3 introduces the methodologies that are used in this study. Section 4 presents the proposed method, which entails a multiple-tile extractor and packet delivery system. Section 5 presents the experimental results and compares the proposed method with existing ones. Section 6 summarizes the conclusions and provides suggestions for future research.

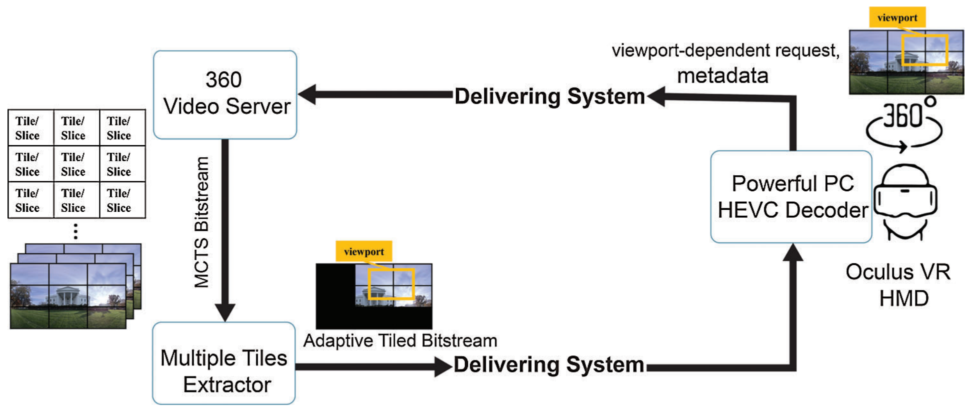

Figure 1: Conceptual architecture of proposed system

2 Related Work

In this section, we discuss prior research on tile-based streaming of 360 videos. Furthermore, research trends with regard to the enhancement of the QoE of 360 VR systems according to user's movements are introduced.

2.1 Tile-Based Streaming of 360 Videos

Currently, 360 VR video streaming services are highly promising; particularly, 360 VR systems that employ tile-based streaming are vital in video optimization. This is because such systems can enable an immersive experience in VR. The solution proposed in [11] is based on tile-based panoramic streaming, whereby users receive a tiles set that match their region of interest (ROI); this solution employs a low-complexity compressed domain video processing technique for using HEVC and HEVC's extensions such as scalable HEVC (SHVC) and multi-view HEVC (MV-HEVC). Additionally, it reduces the peak streaming bitrate under changes in the ROI. This is vital for an immersive experience and low-latency streaming. Furthermore, the solution uses open GOP structures without incurring playback interruptions, thereby providing effective compression, better than that achieved by methods employing closed GOP structures.

In [12], based on the MV-HEVC and SHVC standards, R.G. Youvalari et al. proposed viewport-dependent methods that employ ROI coding for omnidirectional video streaming. The user's viewport is only a part of panoramic videos; thus, to reduce the high bandwidth usage, the part of the scene that corresponds to the user's FOV is transmitted at high quality, and the remainder is delivered at low quality. Hence, their solution can reduce by more than 50% compared to the simulcast method.

In [4,5], the authors presented a novel tile-based streaming solution by transforming 360 videos into mobile VR steams using HEVC and its extension, SHVC. The key idea is that the base layer is used in encoding the entire picture, whereas the enhancement layer is used only for ROI tiles. HEVC and SHVC allow the encoders to encode the bitstream, which can independently transmit tiles. Hence, the generated bitstream is extracted in units of tiles. Based on the HEVC and SHVC standards, the extractor generates the tiled bitstream for the user's viewport. Consequently, the streaming system degrades both computational complexity and network bandwidth. Experimental results proved that this solution could reduce the network bandwidth by up to 47%.

The previous our mobile VR streaming projects contribute 360 video streaming can be approached in a limited performance of mobile VR in [13–17]. Additionally, we conducted studies regarding native VR [1–3]. These experiences enabled us to conduct an improved study on 360 video tile-based streaming for a native VR system connected to a PC based on the advantages of HEVC coding. +. For more details of HEVC's advantages can be reviewed in [18].

2.2 3DoF+/6DoF 360 VR Video

Currently, 3DoF+ and 6DoF provide an immersive experience in VR. However, they require the compression and streaming of multiple videos to support users' body movements, which is considerably challenging with HEVC. As HEVC is designed for single video compression, it requires a large bandwidth and several decoders. Consequently, MV-HEVC [19] was proposed to compress multi-view videos efficiently. MV-HEVC removes the correlation between multiple views at the codec level; moreover, a MV-HEVC decoder can reconstruct the compressed multi-view videos. However, MV-HEVC is not compatible with HEVC; therefore, the existing hardware acceleration employed for HEVC cannot be used for MV-HEVC, and the implementation on mobile devices is difficult. Therefore, in MPEG-I, HEVC (and not MV-HEVC) was employed as the reference software for 3DoF.

In January 2019, proposals on 3DoF+ were obtained with regard to MPEG-I [20]. The corresponding system architecture contains pre-processing and post-processing modules with the existing HEVC codec. To eliminate the correlations among multiple videos, the pre-processing module is included. Moreover, the multiple-video correlation removal process is not carried out at the codec level; therefore, the system can apply the future video codec, versatile video coding (VVC) [21]. In response to the call for proposals on 3DoF+, five proposals [22–26] were submitted in March 2019. Among these, the proposal of Technicolor and Intel [23] demonstrated the best results. Compared with HEVC anchor, this proposal demonstrates a Bjontegaard delta rate (BD-rate) saving of 73.0% for the luma peak signal-to-noise ratio (PSNR) and an average pixel rate ratio saving of 73.34%.

Based on the components of the proposed responses, MPEG-I announced TMIV. Notably, TMIV supports pre-processing and post-processing for streaming multi-view videos to compress 6DoF videos more efficiently. The block diagram of TMIV is presented in Fig. 2. As illustrated in the Fig. 2, TMIV removes the correlations among multiple videos. More details regarding TMIV are provided in [10]. Using the informative areas of the atlases, the TMIV renderer generates a user's viewport. In the last MPEG meeting, several core experiments [27–29] were proposed to improve TMIV.

3 Adaptive VR Streaming—360 Tiled Stream

In this section, we present the proposed method that provides an adaptive viewport-dependent tiled streaming system as shown in Fig. 3. The proposed system consists of three main components: Video client, video streaming server, and packet delivery system. Video client collects the data of movement of the user's head and converting them to metadata (roll, pitch, yaw). It also handles the decoding and rendering tasks. The streaming server encodes original YUV 360 videos into MCTS bitstreams and uses these bitstreams to extract adaptive bitstream according to metadata from video clients. The packet delivery system consists of two components: TCP socket programs and RTSP sender/receiver for exchange request and streaming, respectively. Section 3.1 describes multiple tiles selections on streaming server, and Section 3.2 presents a multiple-tiles extractor. Section 3.3 gives more details regarding the delivery packet using RTSP over TCP. Finally, Section 3.4 identifies several options to reduce motion-to-photon latency.

3.1 Viewport Tile Selection for 360 Video

After a 360 video is transferred to a client, the client decodes the bitstream and forwards the reconstructed video to a renderer. Then, the renderer generates a viewport depending on the user's head movement. From [30], we can verify that equirectangular projection (ERP) implies that if 360 video encoders use a 2D plane video codec, they must project points on the 3D sphere onto a 2D plane video. Yu et al. [30] proposed a method for viewport tile selection for a single 360 ERP video. Based on their outlook, we implemented a viewport multiple-tile selector for a single 360 video.

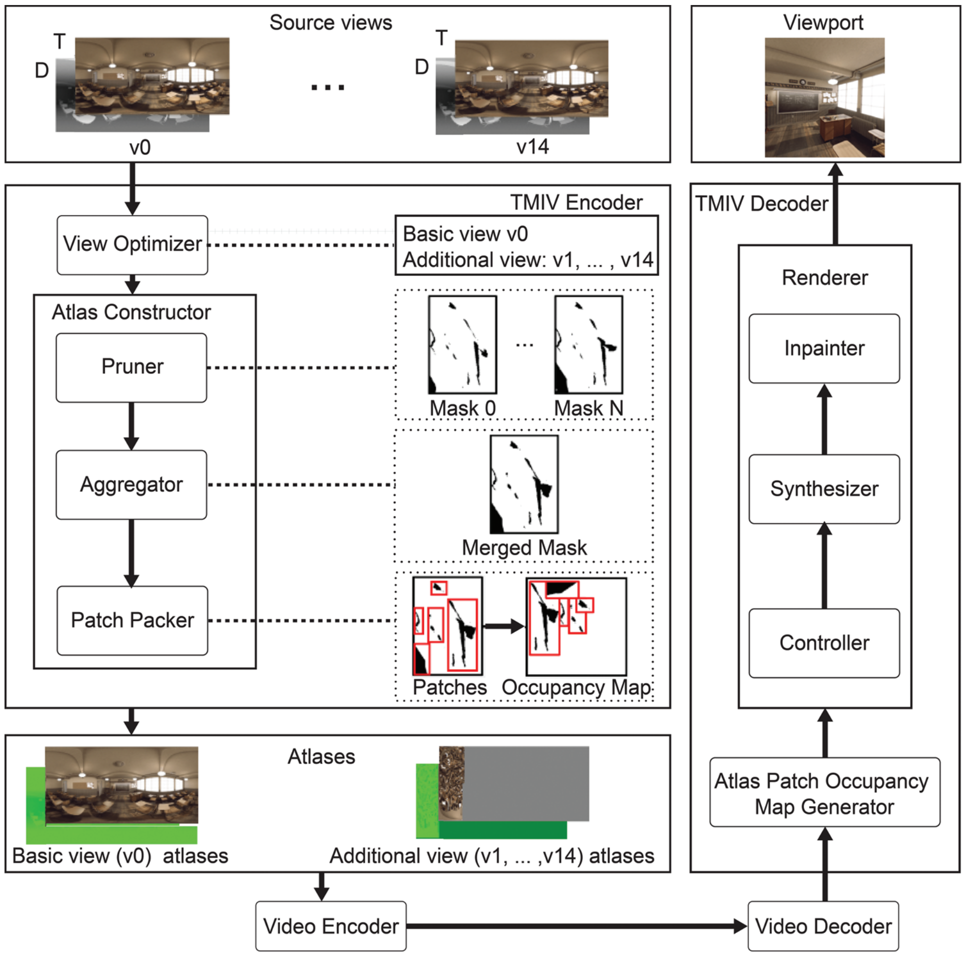

Figure 2: The flow diagram of TMIV

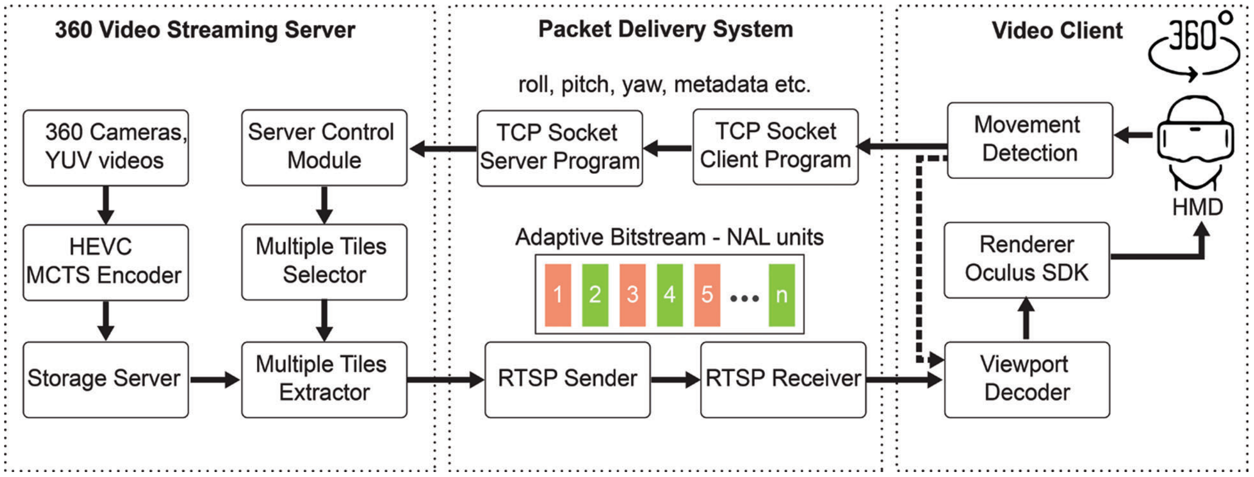

Figure 3: Architecture of viewport-dependent 360 VR streaming system

The rotation of a user's head is represented by a rotation matrix, R, which is equivalent to the user's head being fixed at a regular position with the user looking down in the direction of the negative Z axis. The 3D coordinates are transformed to 2D homogeneous coordinates using a viewport camera intrinsic matrix, K as shown in Eq. (1) below:

$$\mathrm{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

where f_x and f_y denote the focal length of the camera. For instance, let W_{vp} denote the width of the viewport and fov_x denote the horizontal FOV per eye in the HMD. We have $f_x = (W_{vp}/2)(1/tan(fov_x/2))$. c_x and c_y denote the coordinates of the principal point C in the viewport. Let VP denote viewport points on the 360 video, which are represented using Cartesian coordinates, and $vp = [u, v, 1]^T$ denote the 2D homogeneous coordinates of the viewport. Then, VP can be computed as in Eq. (2).

$$VP = R\frac{K^{-1}vp}{\left|\left|K^{-1}vp\right|\right|_2} \tag{2}$$

where denotes the inverse matrix of K and $\left|\left|K^{-1}vp\right|\right|_2$ denotes the L2 norm of $K^{-1}vp$. To obtain the coordinates of the 2D 360 ERP video, we require points in spherical coordinates. A point, VP, in Cartesian coordinates can be converted to a point in spherical coordinates using Eq. (3).

$$\begin{aligned} \phi &= atan2\left(VP_y, VP_x\right)*180/\pi \\ \theta &= asin(VP_z)*180/\pi \end{aligned} \tag{3}$$

The computed spherical coordinates can be converted to the corresponding point in the 2D 360 ERP video using Eq. (4).

$$\begin{aligned} x &= width * (0.5+\phi/360) \\ y &= height * (0.5 + \theta/360) \end{aligned} \tag{4}$$

If the point vp is computed using Eq. (4), a tile that contains vp can be conducted using Eq. (5), where $tile_i$, pic_w, pic_h, $tile_w$, and $tile_h$ represent the tile index, picture width, picture height, tile width, and tile height, respectively.

$$tile_i = (y/tile_h)*(pic_w/tile_w) + x/tile_w \tag{5}$$

In TMIV [10] and OMAF [9], the Euler angle represents the rotation of the user's head using the roll, pitch, and yaw, which correspond to rotation about the X, Y, and Z axes, respectively. As shown in Eq. (6), an angle is generally represented in degrees, and it can be converted into radians to calculate the viewport area. Here, α, β, and γ are angles (in radians) that represent the roll, pitch, and yaw, respectively.

$$\begin{aligned} \alpha &= \alpha_{degree}*\pi/180 \\ \beta &= \alpha_{degree}*\pi/180 \\ \gamma &= \gamma_{degree}*\pi/180 \end{aligned} \tag{6}$$

As shown in Eq. (7), matrix R can be rewritten as the product of the matrices corresponding to rotations about the X, Y, and Z axes.

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\alpha & -sin\alpha \\ 0 & sin\alpha & cos\alpha \end{bmatrix} \begin{bmatrix} cos\beta & 0 & sin\beta \\ 0 & 1 & 0 \\ -sin\beta & 0 & cos\beta \end{bmatrix} \begin{bmatrix} cos\gamma & sin\gamma & 0 \\ sin\gamma & cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (7)$$

The aforementioned method is used to detect viewport tiles of a single 360 video that employs OMAF and TMIV. The proposed method uses this solution to detect the tiles of MCTS bitstreams that are visible in the viewport area. Therefore, the streaming server can exactly determine the tiles that need to be extracted from the original MCTS bitstream.

3.2 Viewport Multiple-Tile Extraction

The latest version (ver. 16.22) of the HEVC reference software (HM) includes an MCTS tile extractor; however, it allows the extraction of only one tile and generates an output bitstream containing only one tile. Hence, the streaming system requires several decoders at the client side for a viewport area that consists of several tiles. Furthermore, aggregating decoded video parts and rendering them on the HMD is time consuming. Therefore, the motion-to-photon latency is extremely high. To solve these problems, we implemented a multiple tile extractor. The extractor selects tiles according to tile indexes, which are determined using viewport tile selection, presented in Eqs. (2) and (5). Next, it extracts the selected tiles from the MCTS bitstream and collates them into a single bitstream, called an adaptive viewport-dependent bitstream.

The HEVC adaptive bitstream consists of a series of network abstraction layer (NAL) units. A detailed explanation regarding the NAL units is presented in [31]. An HEVC bitstream consists of several kinds of NAL units such as parameter sets (VPS), picture parameter sets (PPS), sequence parameter sets (SPS), slices (which are in turn of different types), supplemental enhancement information (SEI), and end of bitstream (EOB) units, as depicted in Fig. 4. The VPS, SPS, and PPS are the most important NAL units because they contain the bitstream information that allows the decoder to interact with subsequent NAL units. A slice consists of a header and a compressed video data field. As depicted in Fig. 5, the multiple-tile extractor parses the input bitstream to read the PPS and obtains the key information: picture size, number of tiles, tile size array, and coding tree unit (CTU) size. In an MCTS-based bitstream, the parameter sets of each tile are stored as extraction information set (EIS) SEI messages.

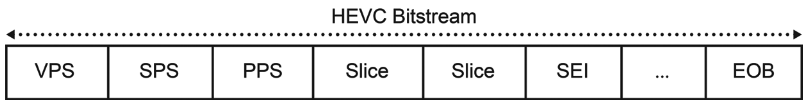

Figure 4: Syntax elements of HEVC bitstream

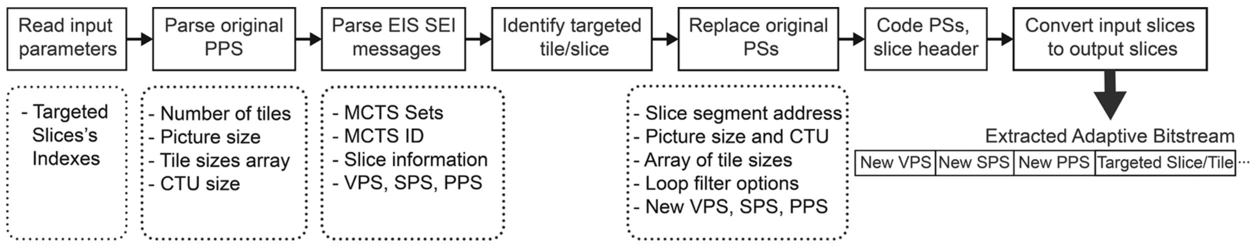

Figure 5: Flow chart of multiple tile extraction

First, the proposed extractor parses the PPS of the input bitstream, and then it parses the EIS SEI messages to acquire the parameter sets (PSs). Next, the extractor identifies the slices that possess the tiles to be extracted. In MCTS-based tiled encoding, one slice contains one tile. According to slices' segment addresses, and the extractor can identify the targeted tile by compared to the input slice's segment address. Thus, the extractor identifies and extracts the targeted slices using the parsed key information. However, the PSs obtained from EIS SEI messages are only for a single tile bitstream. Therefore, to generate the multiple tile bitstream that has the same size as that of the input bitstream, the extractor needs to perform the following special tasks: extracting the targeted slices; replacing certain parameters in the parameter sets, such as picture size, loop filter options, and tiles-enabled flag; encoding the PSs and slice header; and converting the input slices to output slices. Finally, the output adaptive bitstream containing multiple tiles is generated.

The proposed extractor has advantages over the existing single tile extractor. It can reduce the number of required decoders as well as the decoding time. The detailed experimental results of the proposed extractor will be described in Section 4.

3.3 Packet Delivery System

As demonstrated in Fig. 6, to deliver the client request, including metadata such as roll, pitch, and yaw data, we implemented a packet delivery system-based TCP socket. The TCP socket program is reasonable for low metadata traffic. Additionally, we implemented the video stream delivery system using RTSP over TCP. However, RTSP has a limitation in that it provides high performance only for a single connection, which is aimed at serving one user at a time. Additionally, we considered MPEG/DASH [32] or HLS [33], which can also be applicable to the proposed system. However, they perform accurately only for video sources such as MP4 files. An MCTS bitstream has certain restrictions such as the packet overhead problem, high latency, and high computational complexity. Therefore, rather than using RTSP in a centered model, we re-designed the RTSP/TCP delivery system from a centered model to a distributed one, as demonstrated in Fig. 6. Here, a video client can open a listening RTSP session on a specified port, Port$_{rec}$, when it initializes the transmission of a viewport-dependent request to a video server. The metadata sent to the server includes roll, pitch, yaw, other viewport metadata, and Port$_{rec}$. According to the request, the streaming server forwards the adaptive viewport-dependent bitstream accurately to the video client via a special RTSP link in the format "rtsp://video_client_IP_address: Port$_{rec}$". To solve the problem when the network failed, the server and client control modules will handle to reinitialize a new session. Using this method, a streaming server can serve multiple clients simultaneously.

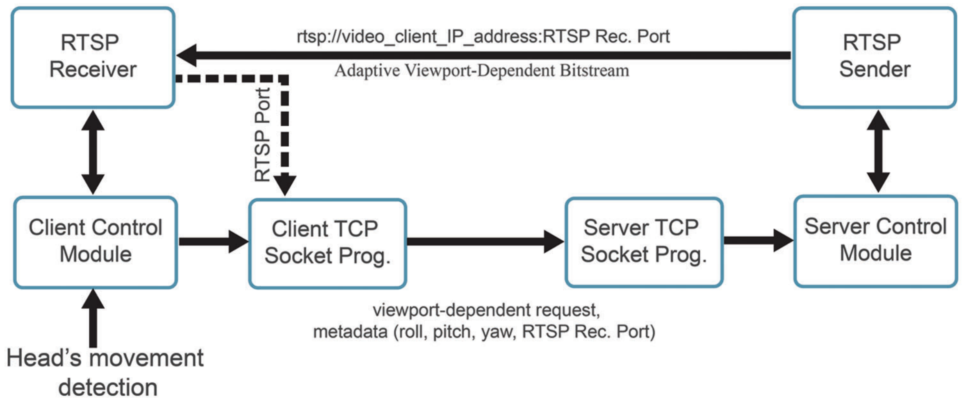
RTSP Receiver
rtsp://video_client_IP_address:RTSP Rec. Port
Adaptive Viewport-Dependent Bitstream
RTSP Sender
RTSP Port
Client Control Module
Client TCP Socket Prog.
Server TCP Socket Prog.
Server Control Module
viewport-dependent request,
metadata (roll, pitch, yaw, RTSP Rec. Port)
Head's movement detection

Figure 6: Flow chart of packet delivery system

3.4 Reducing Motion-to-Photon Latency

Motion-to-photon latency is defined as the time delay between a user's initial head movement and the rendering of the first image on the HMD. As illustrated in Fig. 7, the user's head movement will cause the occurrence of a "user viewport change event." The video client will make an event of viewport change, and then it creates a chunk request according to the viewport data. The motion-to-photon latency can be computed as given in Eq. (8).

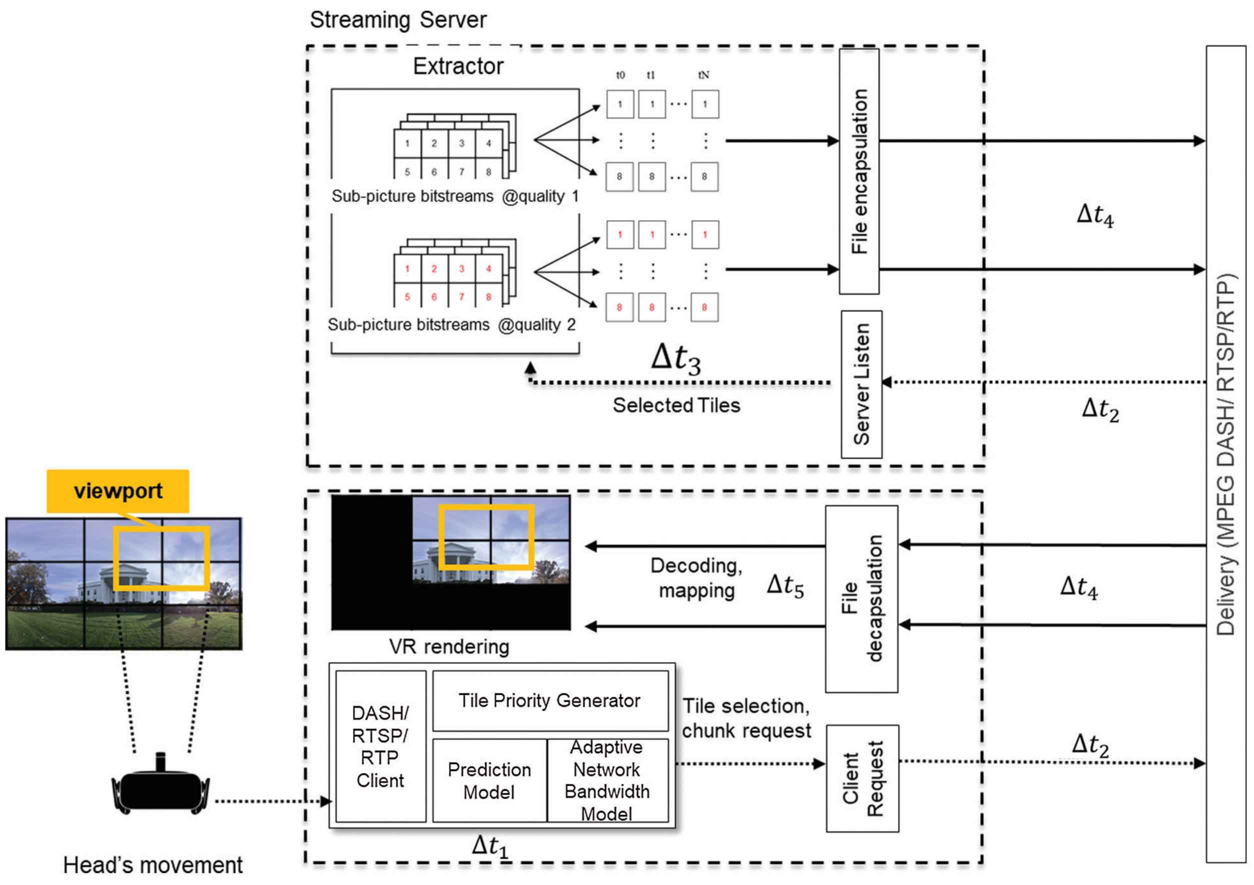

Figure 7: Motion-to-photon latency of the proposed system

$$\text{Motion\_to\_photon\_latency} = \Delta t_1 + \Delta t_2 + \Delta t_3 + \Delta t_4 + \Delta t_5 \quad (8)$$

where Δt_1 denotes the processing latency between a user's head movement and the client sending a viewport change request; Δt_2 the request latency; Δt_4 the transmission latency; Δt_3 the processing latency of the streaming server between receiving the request and transmitting an adaptive bitstream to the delivery system; and Δt_5 the processing latency entailed in decoding and rendering a user's viewport on the VR HMD. Δt_2 depends on the size of the metadata and client request. Therefore, to reduce Δt_2, we must substantially decrease the size of the client request.

Additionally, the performance of the delivery packet system based on the TCP socket, as described in Section 3.3, affects Δt_2. Based on the factors that affect the motion-to-photon latency, we propose the following approaches to reduce the total latency: Optimizing tile size according to network bandwidth condition; Using a prediction model to predict the movement of user's head, then client can generate bitstream request faster than before; Streaming both high-quality bitstream and low-quality bitstream to client. These options can be described in sub-sections in below.

3.4.1 Optimization of Tile's Size

As illustrated in Fig. 7, to reduce the motion-to-photon latency, one approach is to reduce the workload of the proposed system. From the network bandwidth point of view, we verified that at a specified network bandwidth, the tile size of an adaptive bitstream can affect Δt_4. Furthermore, a small tile size reduces the processing time Δt_3 in the extraction of the adaptive bitstream. Moreover, a smaller viewport area decreases the number of decoding and rendering operations. This means that the size of tile can affect Δt_5. An adaptive network bandwidth model has been implemented to identify the network bandwidth condition at the video client.

3.4.2 The Prediction for User's Head Movement

This prediction model is based on the notion that the video client can predict viewport tiles that are required for an adaptive bitstream according to changes in the coordinates of eyes and the speed of the user's head movement. This reduces the processing latency Δt_1.

3.4.3 High-Quality and Low-Quality 360 Streams

Rather than requesting a new chunk, the video client employs low-quality tiles corresponding to the same location. Because the size of low-quality tiles is small, the video client can decrease the processing time Δt_1. Additionally, without making a new request, the proposed system can reduce latencies Δt_3, Δt_4, and Δt_5. As shown in Fig. 7, the extractor on streaming server parsed encoded bitstreams with quality 1 as high-quality video, and quality 2 as low-quality video. Thus, extractor can generate adaptive tiled bitstream in various qualities of 360 videos according to details of selected tiles.

4 Experimental Results

4.1 Testbed Scenario

To test the performance of the proposed system, we built a test environment. We set up a streaming server with the following configuration: Intel Xeon E5-2687W v4 CPU (24 cores, 48 threads total); 128 GB of memory; a GTX 1080 Ti GPU; and an Ubuntu 64 bit (gcc 6.3 v. 18.04) operating system. Further, a video client PC with a Core i7-7700 4-core 8-threaded 4.2 GHz CPU, one Nvidia GeForce 1080 GPU, and 32 GB of memory with the Windows 10 operating system was employed. Both Oculus Rift and Rift S were used as HMDs with Oculus SDK version 1.43. Additionally, the network environment was installed using the internal network of Sungkyunkwan University. The mandatory parameters of the experiments are presented in Tab. 1. The following original 4K videos are employed as standard 360 video test sequences [34]: AerialCity, DrivingInCity, DringInCountry, and PoleVault_le, with the coding parameters set as presented in Tab. 2.

We used HM software with 360 libraries [35] as a video encoder to produce low-quality and high-quality bitstreams using ERP. The libav-ffmpeg library [36] was employed to implement a fast video decoder with an RTSP receiver. We used a method called weighted in sphere PSNR to calculate the distortion in the sphere to verify the quality of the reconstructed 360 videos. To compute the dissimilarity between the reconstructed 360 video and original video, sphere PSNR or WS-PSNR uses a weighted metric to determine the distortion in the spherical domain. FFmpeg tools [37], WS-PSNR software [38], VMAF [39], and IV-PSNR [40] were used as evaluation tools. More details on IV-PSNR and VMAF can be obtained via standard documents regarding MPEG-I [41].

Table 1: System parameters for experiments

Parameters	Values
QP	22, 27, 32, 37, 42
GOP	16
Framerate	30 fps
Intraperiod	32
FOV	$90^{o} \times 90^{o}$
Evaluation frames	90 frames
Segment duration	30 frames

Table 2: JCT-VC test sequences in detail

Parameters	Values
Input bit depth	8 bits
Input chroma format	420
Frame rate	30 fps
Frame skip	0
Width in pixels	3840
Height in pixels	1920
Number of frames	300
Coding level	5.2

The test sequences were encoded using five quantization parameters (QPs). The first four QP values were employed to encode the tile layer (high quality), and the last QP value was used to encode the base layer (low quality). The group-of-pictures (GOP) value was set to 16, and the framerate was 30 fps. The FOV of the viewport was 90° × 90°, and the evaluation frames were 90 frames that were partitioned into smaller parts. Considering the streaming scenario, the videos were divided into 30 frame chunks. The sum of viewport tiles of the chunk frames was selected. The viewport tiles were computed using the proposed tile selection, based on Eq. (5); furthermore, they were extracted from the MCTS bitstream using the proposed multiple tile extractor. At the client PC, the libav-ffpmeg library (with NVIDIA GPU acceleration [42]) was employed to implement the fast HEVC decoder. After decoding, the decoder generated the user's viewport using the reconstructed video. Finally, the viewport was rendered on an Oculus Rift S using our VR program, which was implemented based on the Oculus PC SDK [43]. To generate the viewport, the user's movement data were required. We considered the scenario of the user's movement in a fixed direction. For instance, the viewport setting for the movement scenario [90.00 90.00 90.00 0] implies that the horizontal and vertical FOVs were set to 90^{o} and 90^{o}, respectively, and the center of the viewport was set at a 90^{o} longitude and 0^{o} altitude.

4.2 Performance Evaluation

In tiled 360 video streaming, the tile size directly affects the performance of the streaming service with regard to several aspects such as decoding time and resource usage of the client [44]. To determine a reasonable tile size for the proposed system, we performed various experiments using three tile sizes, 320 × 320, 640 × 640, and 960 × 960, on two test sequences, AerialCity and PoleVault_le. See Tab. 3, the result demonstrates that with a small tile size, 320 × 320, the proposed system can achieve the best BD-rate saving of 25.89% for Y-PSNR. Tile sizes of 640 × 640 and 960 × 960 yield BD-rate savings of 19.25% and 10.35%, respectively. This is because for 3DoF+ 360 videos or traditional 360 videos, a smaller tile size can yield a better encoding efficiency. Additionally, this tile size can provide the highest numbered results in VMAF, MS-SSIM, and IV-PSNT tests. Fig. 8 depicts the rate-distortion (RD) curves of the non-tiled and proposed tiled streaming methods for different tile sizes. As depicted in the figure, the proposed tiled streaming method provided better results compared to those obtained via non-tiled streaming. Among the test results obtained for various tile sizes, a tile size of 320 × 320 provides the best results; the basic views of the tile layer are encoded using the tile size of 320 × 320.

Table 3: Performance of the proposed tiled streaming method for various tile sizes

Tiled Stream	Y-PSNR (%)	VMAF (%)	MS-SSIM (%)	IV-PSNR (%)
320 × 320	−25.89	−13.92	−18.84	−25.54
640 × 640	−19.25	−9.62	−14.47	−17.91
960 × 960	−10.35	2.53	−2.13	−6.61
Average	−18.49	−7.0	−11.81	−16.69

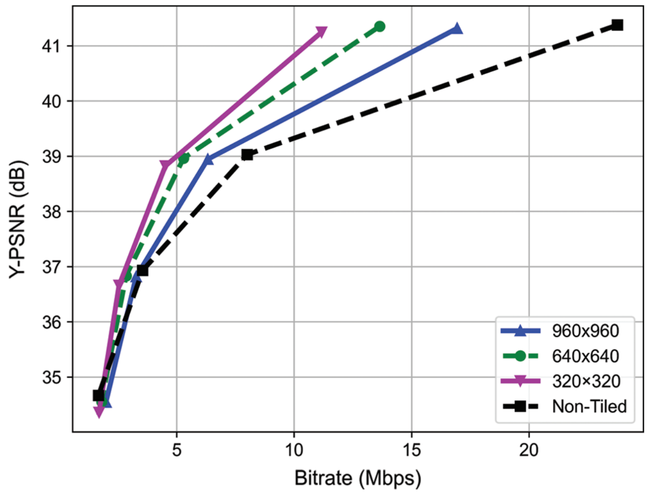

Figure 8: RD-curves for the non-tiled streaming and proposed tiled streaming methods on tile sizes

Fig. 9 presents the maximum memory usage under decoding and the decoding time for both the single tile extractor and proposed multiple tile extractor. By partitioning 360 videos into 12 × 6 grid tiles, the single tile extractor consumed 16.18 GB of memory and 13.2 s for decoding the viewport tiles, whereas the multiple tiles extractor used only 3.1 GB of memory and 2.06 s, that is, 0.023 s per frame. Moreover, the multiple tile extractor exhibited nearly similar decoding memory usage and time consumption for the three tile sizes. Therefore, grid-tile partitioning can be employed to reduce the bandwidth without considerably affecting the decoding resources. The required decoding time can be reduced when further optimization, such as parallel tile decoding on HEVC codec embedded hardware, is conducted. Therefore, the advantage of using the multiple tile extractor is verified, and we have used this for the overall experiment.

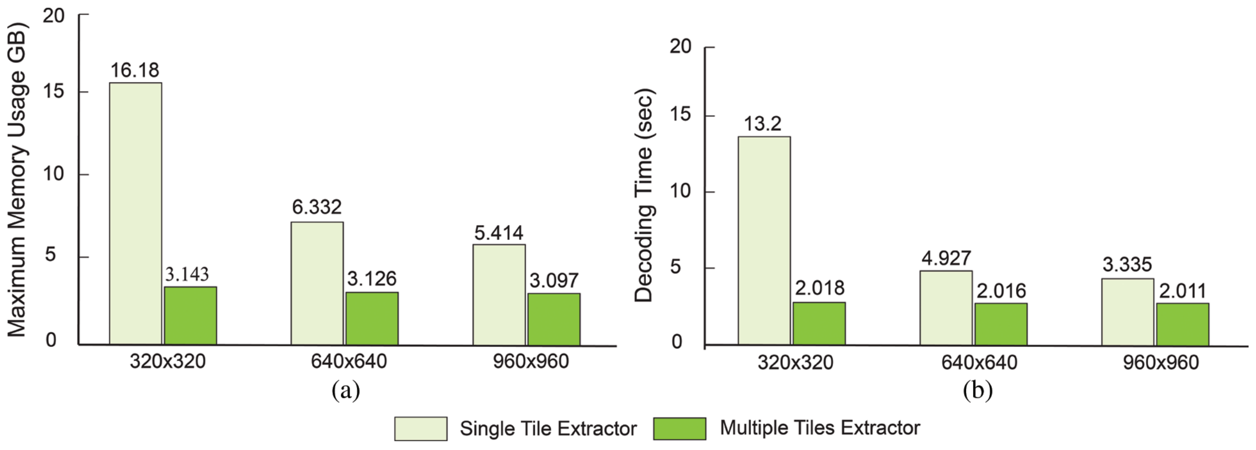

Figure 9: Performance comparison of the single and multiple tile extractors regarding (a) maximum memory use and (b) decoding time

As presented in Tab. 4, the proposed method can achieve an average bitrate saving of 34.56%. The bitrate saving is the highest for test-sequence AerialCity because the viewport area in AerialCity is at the bottom- right corner as depicted in Fig. 10b. This test-sequence includes many objects that are very realistic and moving, including the light effect on the scene for all of the tiles, and the bitrates that are required for several tiles are almost similar. However, other test sequences have the most complex objects in the view area and the tiles except the view only require a small amount of bitrate. For instance, the viewport area for PoleVault_le depicted in Fig. 10c, the viewport is in the most complicated portion of the scene compared to other test-sequences. Consequently, the viewport of PoleVault_le test-sequence acquired the lowest bitrate saving. Additionally, Tab. 5 presents the BD-rate savings of the proposed tiled streaming method compared to those of non-tiled streaming. With regard to luma PSNR, VMAF, and IV-PSNR, it provides an average BD-rate savings of 25.89%, 13.92%, and 25.24%, respectively. Finally, from the outcomes of the proposed system, we confirmed that it also can be applied to 6DoF 360 video streaming by warping multiple 360 video for multiple tiles selection.

Table 4: Bitrate savings of the proposed method

Bitstream	Bitrate savings (%)
AerialCity	−38.42
DrivingInCity	−35.72
DrivingInCountry	−36.22
PoleVault_le	−27.88
Average	−34.56

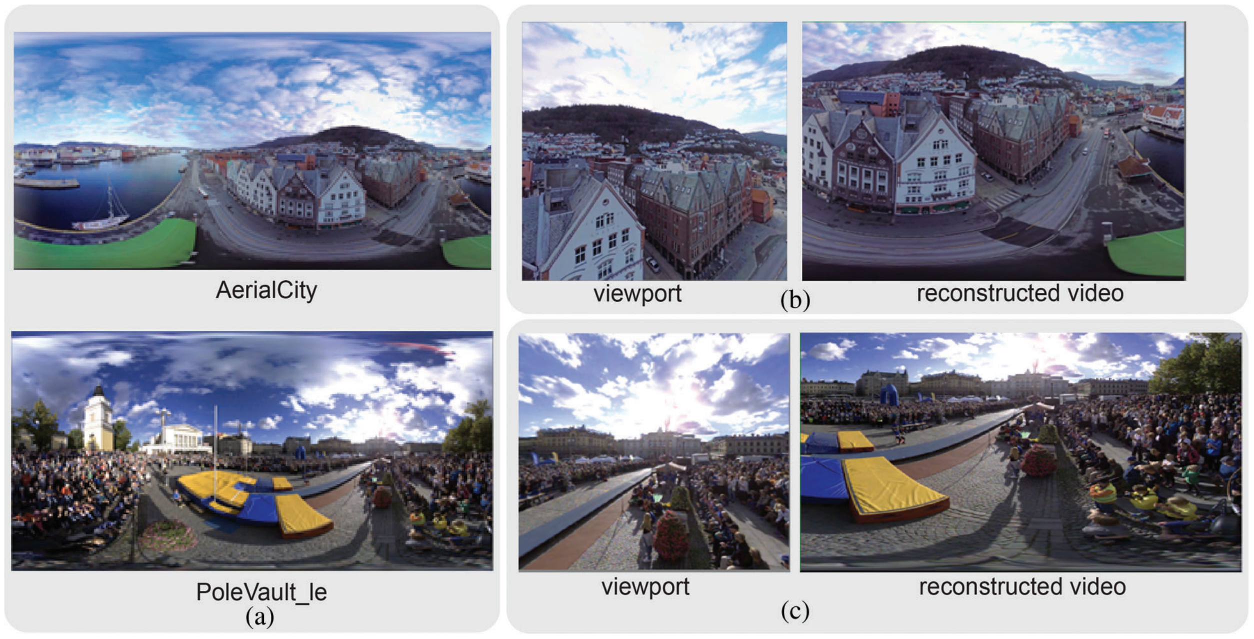

Figure 10: Output of the proposed system for (a) video source AerialCity and PoleVault_le; user's viewport and reconstructed video for (b) AerialCity and (c) PoleVault_le

Table 5: BD-rate savings of the proposed tiled streaming method

Tiled Stream	Y-PSNR (%)	VMAF (%)	MS-SSIM (%)	IV-PSNR (%)
AerialCity	−19.95	−7.42	−13.13	−24.96
DrivingInCity	−29.78	−21.68	−26.17	−27.71
DrivingInCountry	−29.58	−20.52	−17.85	−24.32
PoleVault_le	−24.25	−6.05	−18.21	−25.17
Average	−25.89	−13.92	−18.84	−25.54

5 Conclusion

Herein, we proposed an adaptive 360 streaming solution for VR systems. The proposed approach allowed the server to analyze the user's viewport-dependent details and generate a unique bitstream that includes the tiles that comprise the viewport area. The proposed system consisted of a viewport tile selector and an extractor that allowed multiple-tile processing. Additionally, the areas corresponding to the user's viewport were selected, and the result could be extended for 6DoF 360 streaming by warping multiple videos. The proposed extractor extracted all selected tiles from the MCTS source-view bitstream and collated them into to one adaptive bitstream; then, the latter bitstream was transmitted to the client via a packet delivery system. Furthermore, we suggested approaches for reducing the motion-to-photon latency. In this regard, we performed an optimization based on the tile size. We found that a tile size of 320 × 320 was reasonable for the proposed 360 video-based tiled streaming system. The proposed method demonstrated BD-rate savings of 25.89% for the Y-PSNR and decoding time savings of 61.16% compared to previous non-tiled streaming methods or other methods that used a single tile extractor. Furthermore, we plan to implement two other approaches to reduce the motion-to-photon latency in the future. Moreover, an important objective for future work is the implementation of a real-time 6DoF tiled streaming system based on the proposed method.

Funding Statement: This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-00765, Development of Compression and Transmission Technologies for Ultra High-Quality Immersive Videos Supporting 6DoF).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] J. B. Jeong, D. M. Jang, J. W. Son and E. S. Ryu, "3DoF+ 360 video location-based asymmetric down-sampling for view synthesis to immersive VR video streaming," *Sensors*, vol. 18, no. 9, pp. 3148–3167, 2018.

[2] J. B. Jeong, D. M. Jang, J. W. Son and E. S. Ryu, "Bitrate efficient 3DoF+ 360 video view synthesis for immersive VR video streaming," in *2018 Int. Conf. on Information and Communication Technology Convergence, IEEE*, JeJu Island, Korea, pp. 581–586, 2018.

[3] J. B. Jeong, S. B. Lee, D. M. Jang and E. S. Ryu, "Towards 3DoF+ 360 video streaming system for immersive media," *IEEE Access*, vol. 7, pp. 136399–136408, 2019.

[4] J. W. Son, D. M. Jang and E. S. Ryu, "Implementing 360 video tiled streaming system," in *Proc. of the 9th ACM Multimedia Systems Conf. ACM*, Amsterdam, Netherlands, pp. 521–524, 2018.

[5] J. W. Son and E. S. Ryu, "Tile-based 360-degree video streaming for mobile virtual reality in cyber physical system," *Computers & Electrical Engineering*, vol. 72, pp. 361–368, 2018.

[6] R. Skupin, Y. Sanchez, K. Sühring, T. Schierl, E. S. Ryu *et al.,* "Temporal MCTS coding constraints implementation," in *122th MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, San Diego, CA, USA, m42423, 2018.

[7] M. L. Champel, T. Stockhammer, T. Fautier, E. Thomas and R. Koenen, "Quality requirements for VR," in *116th MPEG Meeting of ISO/IEC JTC1/SC29/WG11*, Chengdu, China, m39532, 2016.

[8] S. Deshpande, Y. K. Wang, M. M. Hannuksela and S. Deshpande, "Preliminary WD of ISO/IEC 23090-2 2nd edition OMAF JTC1/SC29/WG11," in *127th MPEG Meeting of ISO/IEC JTC1/SC29/WG11*, Gothenburg, Stockholm, pp. 23090–23092, 2019.

[9] Y. K. Wang, "An overview of OMAF (for information)," in *121st MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, Gwangju, Korea, m41993, 2017.

[10] B. Salahieh, B. Kroon, J. Jung and M. Domański, "Test model 3 for immersive video," in *128th MPEG Meeting of ISO/IEC JTC1/SC29/WG11*, Brussels, Belgium, n18795, 2019.

[11] Y. Sánchez de la Fuente, R. Skupin and T. Schierl, "Video processing for panoramic streaming using HEVC and its scalable extensions," *Multimedia Tools and Applications*, vol. 76, no. 4, pp. 5631–5659, 2017.

[12] R. G. Youvalari, M. M. Hannuksela, A. Aminlo and M. Gabbouj, "Viewport-dependent delivery schemes for stereoscopic panoramic video," in *2017 3DTV Conf.: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON)*, Copenhagen, Denmark, pp. 1–4, 2017.

[13] Y. I. Ryu and E. S. Ryu, "Video on mobile CPU: UHD video parallel decoding for asymmetric multicores," in *Proc. 8th ACM Int. Conf. Multimedia Syst. (MMSys)*, Taipei, Taiwan, pp. 229–232, 2017.

[14] J. W. Son, D. M. Jang and E. S. Ryu, "Implementing motion constrained tile and viewport extraction for VR streaming," in *Proc. of Network and Operating System Support on Digital Audio and Video Workshop (NOSSDAV)*, Amsterdam, Netherland, pp. 61–66, 2018.

[15] H. W. Kim, T. T. Le and E. S. Ryu, "360-degree video offloading using millimeter-wave communication for cyberphysical system," *Transactions on Emerging Telecommunications Technologies*, vol. 30, no. 4, e3506, 2018.

[16] T. T. Le, D. N. Van and E. S. Ryu, "Real-time 360-degree video streaming over millimeter wave communication," in *Proc. of Int. Conf. on Information Networking*, Chiang Mai, Thailand, pp. 857–862, 2018.

[17] T. T. Le, D. N. Van and E. S. Ryu, "Computing offloading over mmWave for mobile VR: Make 360 video streaming alive," *IEEE Access*, vol. 6, pp. 66576–66589, 2018.

[18] T. Rahum and S. Y. Shin, "Subjective evaluation of ultra-high definition (UHD) videos," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 6, pp. 2464–2479, 2020.

[19] M. M. Hannuksela, Y. Yan, X. Huang and H. Li, "Overview of the multi-view high efficiency video coding (MV-HEVC) standard," in *2015 IEEE International Conference on Image Processing*, Quebec, Canada, pp. 2154–2158, 2015.

[20] ISO/IEC JTC1/SC29/WG11, "Call for proposals on 3DoF+ visual," in *125th MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, Marrakesh, Morocco, pp. n18145, 2019.

[21] J. R. Ohm and G. J. Sullivan, "Versatile video coding-towards the next generation of video compression," in *Picture Coding Symp. 2018*, San Francisco, CA, USA, 2018.

[22] M. Domanski, A. Dziembowski, D. Mieloch, O. Stankiewicz, J. Stankowski *et al.,* "Technical description of proposal for call for proposals on 3DoF+ visual prepared by Poznan university of technology (PUT) and Electronics and telecommunications research institute (ETRI)," in *126th MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, Geneva, Switzerland, m47407, 2019.

[23] J. Fleureau, F. Thudor, R. Dore, B. Salahieh, M. Dmytrychenko *et al.,* "Technicolor-intel response to 3DoF+ cfp," in *126th MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, Geneva, Switzerland, m47445, 2019.

[24] B. Kroon and B. Sonneveldt, "Philips response to 3DoF+ visual cfp," in *126th MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, Geneva, Switzerland, m47179, 2019.

[25] V. K. M. Vadakital, K. Roimela, L. Ilola, J. Keränen, M. Pesonen *et al.,* "Description of Nokia's response to cfp for 3DOF+ visual," in *126th MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, Geneva, Switzerland, m47372, 2019.

[26] B. Wang, Y. Sun and L. Yu, "Description of Zhejiang University's response to 3DoF+ visual cfp," in *126th MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, Geneva, Switzerland, m47684, 2019.

[27] H. H. Kim, Y. U. Yoon, J. G. Kim, G. S. Lee, J. Y. Jeong *et al.,* "MPEG-I visual CE3-related: CTU-based packing," in *128th MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, Geneva, Switzerland, m49055, 2019.

[28] S. B. Lee, J. B. Jeong and E. S. Ryu, "CE3: Viewport-dependent patch grouping using HEVC tiles," in *128th MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, Geneva, Switzerland, m50818, 2019.

[29] B. Salahieh and J. Boyce, "Intel response to immersive video CE1: Group-based TMIV," in *128th MPEG Meeting of ISO/IEC JTC1/SC29/ WG11*, Geneva, Switzerland, m49958, 2019.

[30] M. Yu, H. Lakshman and B. Girod, "A framework to evaluate omnidirectional video coding schemes," in *2015 IEEE Int. Sym. on Mixed and Augmented Reality*, Fukuoka, Japan, pp. 31–36, 2015.

[31] V. Sze, M. Budagavi and G. J. Sullivan, *High Efficiency Video Coding*. Switzerland: Springer, 2014. [Online]. Available: https://www.springer.com/gp/book/9783319068947.

[32] N. Kim and B. D. Lee, "Analysis and improvement of MPEG-DASH-based internet live broadcasting services in real-world environments," *KSII Transactions on Internet and Information Systems*, vol. 13, no. 5, pp. 2544–2557, 2019.

[33] Ed R. Pantos, "RFC 8216: HTTP live streaming specification document, Cupertino, CA, USA," 2017. [Online]. Available: https://developer.apple.com/streaming/.

[34] J. R. Ohm and G. Sullivan, *Joint Collaborative Team on Video Coding Standard Development*. Kyoto, Japan: ITU, 2010. [Online]. Available: https://www.itu.int/dms_pub/itu-t/oth/46/01/T46010000010001PDFE.pdf.

[35] ITUT SG16 WP3 & ISO/IEC JTC1/SC29/WG11, "Ver. 16.22, HM-high efficiency video coding," 2020. [Online]. Available: https://vcgit.hhi.fraunhofer.de/jct-vc/HM.

[36] F. Bellard, "Ver. 4.3.1, FFMPEG library audio video," 2020. [Online]. Available: https://github.com/FFmpeg/FFmpeg.

[37] F. Bellard, "Ver. 4.2, FFmpeg tools," 2020. [Online]. Available: https://www.ffmpeg.org/.

[38] MPEG Explorations 3DoFplus, "Ver. 2.0.1, The ERP WS-PSNR software," 2019. [Online]. Available: http://mpegx.int-evry.fr/software/MPEG/Explorations/3DoFplus/ERP_WS-PSNR.

[39] Netflix Inc., "Ver. 1.5.2, The VMAF software," 2020. [Online]. Available: https://github.com/Netflix/vmaf.

[40] MPEG-I, immersive video, MPEG 127–Gothenburg, "Ver. 2.0, The IV-PSNR evaluation tool for immersive video," 2020. [Online]. Available: https://gitlab.com/mpeg-i-visual/ivpsnr.

[41] A. Dziembowski, "Software manual of IV-PSNR for immersive video," in *128th MPEG meeting of ISO/IEC JTC1/SC29/ WG1*, n18709, 2019.

[42] NVIDIA Corporation, "Ver. 4.3, using FFmpeg with NVIDIA GPU hardware acceleration," 2020. [Online]. Available: https://developer.nvidia.com/ffmpeg.

[43] Facebook Inc., "Ver. 1.43, The oculus PC SDK for windows," 2020. [Online]. Available: https://developer.oculus.com/downloads/package/oculus-sdk-for-windows/.

[44] A. Zare, A. Aminlou, M. M. Hannuksela and M. Gabbouj, "HEVC-compliant tile-based streaming of panoramic video for virtual reality applications," in *Proc. of the 24th ACM Int. Conf. on Multimedia*, Amsterdam, Netherland, pp. 601–605, 2016.