J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr

DASH 기반 자유시점 비디오 스트리밍 시스템 구현☆

Implementation Method for DASH-based Free-viewpoint Video Streaming System

서 민 재[1] 백 종 호[2*]
Minjae Seo Jong-ho Paik

ABSTRACT

Free-viewpoint video (FVV) service provides multi viewpoints of contents and synthesizes intermediate video files which are not captured on some view angles so that enables users to watch as they choose wherever they want. Synthesizing video is necessary technique to provide FVV video service, because every video of the FVV contents for different view angles cannot be stored to the content server physically. For the reason, fast view synthesis can improve the quality of video service and increase user's satisfaction. One of the studies for FVV service, a method was proposed to transmit FVV service based on DASH (Dynamic Adaptive Streaming over HTTP). There is big advantage on using DASH that it is commonly used to transport video service. However, the method was only a conceptual proposal, so it is difficult to implement the system using the proposal.

In this paper, we propose an implementation method to provide real-time FVV service smoothly. We suggest a system structure and operation method on the server and client side in detail, which is to be applicable to synthesize video quickly. Also, we suggest generating FVV service map additionally which controls a FVV service overall. We manage real-time information of the whole service through the service map. The service can be controlled by reducing the possible delay from network situation.

☞ keyword : Free-viewpoint Video, DASH, Multi-viewpoint, FVV, Streaming Service, Immersive Video

1. Introduction

With the improvement of image processing technology and hardware device specifications, the way to provide video is gradually changing from a passive mode which shows a screen with only one fixed view. To allow users to watch a video immersively, video services are being provided in various forms recently such as VR (Virtual Reality), AR (Augmented Reality) and so on. For users' immersive experience, FVV service has been studied actively in video experts groups and video service providers[1]. FVV service provides multi viewpoints of a video contents and synthesizes intermediate video files which are not captured on that view angles so that enables users to watch as they choose wherever they want.

The technique of synthesizing video is necessary in order to provide FVV video service, because the service contents physically cannot be stored to the contents server with every video file for different viewpoints. Also, synthesizing view is very important to for providing stable video service which is directly connected to QoS (Quality of Service). Synthesizing view can be processed on both server and client side, but generally FVV services are required to synthesize on the client side for smooth playback considering the existing computer capacity.

To provide FVV streaming, one method was proposed to transport FVV video which synthesizes video on the client side based on DASH [2]. There is big advantage on using DASH that it is commonly used to transport video service. Contents providers can easily apply the method to the existing systems, and users can be serviced without particular

1 Dept. of Computer, Seoul Women's University, Seoul, 01797, Korea.
2 Dept. of Software Convergence, Seoul Women's University, Seoul, 01797, Korea.
* Corresponding author (paikjh@swu.ac.kr)
[Received 31 August 2018, Reviewed 30 October 2018, Accepted 05 December 2018]
☆ This work was supported by a special research grant from Seoul Women's University(2018).
☆ A preliminary version of this paper was presented at ICONI 2017 seminar and was selected as an outstanding paper.

restriction. However, since the method is only a conceptual proposal, it is difficult to implement the system using it.

Therefore, we propose an implementation method to provide real-time FVV service smoothly in this paper. We suggest the system structure and operation method on the server and the client side both to synthesize video quickly. It makes possible to provide FVV service stably as well. In addition, we suggest how to send a service map additionally which controls a FVV service overall, so the synthesized video and existing video can be mapped according to the user's viewpoint. This mapping information can reduce the delay that may occur depending on the network environment. It helps to find nearest video which can be serviced with the circumstances when users want to change views and it helps to maintain seamless FVV service. Through the proposed implementation method, the service provider can provide free-viewpoint service stably.

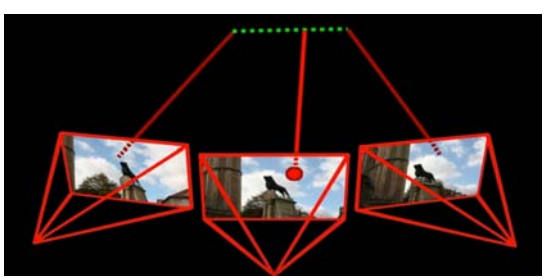
2. Related Work

2.1 FVV (Free-Viewpoint Video)

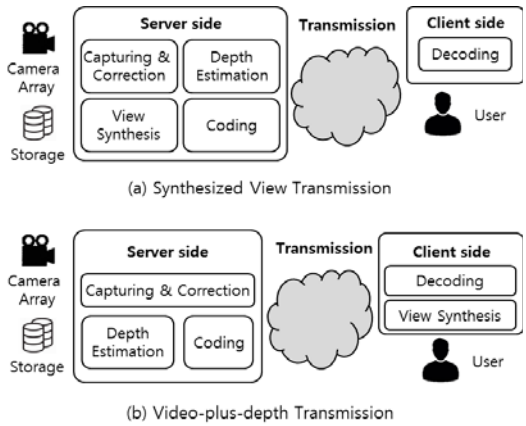
Free-viewpoint video is interactive multimedia service that allows users to change their viewpoint or directions, and search whatever they want to watch[3]. There are many studies for provide video service with multi viewpoints.

The video service is efficient for the sport games and concerts for amusement and many contents provider is trying to service FVV.

It is necessary to synthesize video which is not captured by camera, but there is still problem that it takes much time to be serviced and the method is not simple to be performed.



(Figure 1) Correspondence and Depth-Image Based Rendering for free-viewpoint video[4]



(Figure 2) Free-Viewpoint video transmission Models

2.2 A DASH based Free-viewpoint Video Streaming System

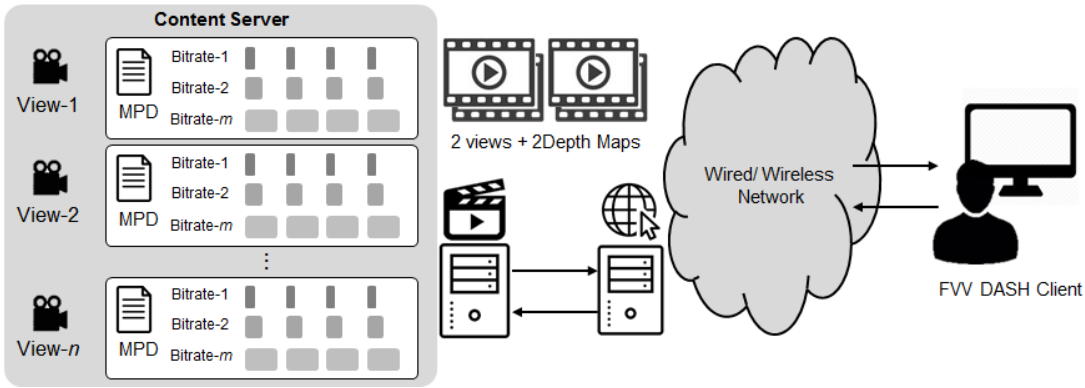
Ahmed proposed DASH-based FVV streaming system[2]. Free-viewpoint video streaming method can be divided into two kinds.

First, when a client requests viewing at a time when the original image does not exist, the server synthesizes the intermediate viewpoint and sends the corresponding data. Second one is a method that the service provider provides necessary data in advance such as video and depth information for composition, and the client synthesizes the video with using the corresponding information and plays the video[2].

In the case of Ahmed, the latter study was conducted. It is noted that processing time can be reduced by processing the video processing technique such as video synthesis at the client side, and a higher quality image can be transmitted within a given bandwidth. In addition, because the network environment of each user can be different, it is possible to select the video quality variably, thereby preventing QoS degradation due to service disconnection [4].

This is largely related to the four main characteristic of FVV[2].

- Responsiveness - The user must receive a response from the system in real time, and the delay time from



(Figure 3) Architecture of the free-viewpoint video streaming service

acquiring the object after requesting the change of view should be minimized. It includes both network related delays and processing delays.

- Scalability - It should be able to process smoothly even when multiple clients connected at the same time to request a screen of another view.
- Adaptability - It should be able to provide optimal quality to clients over heterogeneous network even network environments change dynamically.
- Immersiveness - The user must be able to choose any points of view, and smooth and responsive video services should be provided.

To solve this problem, he tried to manage a large number of users flexibly using an HTTP web server. It enables quick response to the user's requests while changing the video quality according to the view synthesis function and the network bandwidth on the client side. The streaming system architecture proposed by Ahmed is shown in figure 3.

There are a content server and a web server in the server side. Depending on the viewpoint position, each video is segmented by time MPD (Media Presentation Description) document manages this video address. MPD document can exist in three forms. First, there is a MPD per stream, and a MPD exists for each viewpoint. Finally, there is a MPD that manages the entire service. Ahmed assumes that there is a metafile that manages the entire MPD, original video and depth video components. The client side can parse the metafile and download the video to receive the service.

Ahmed also proposed bandwidth-based switching logic based on sampled R-D (rate-distortion) values, but this paper does not consider the part and focuses on the overall streaming service structure.

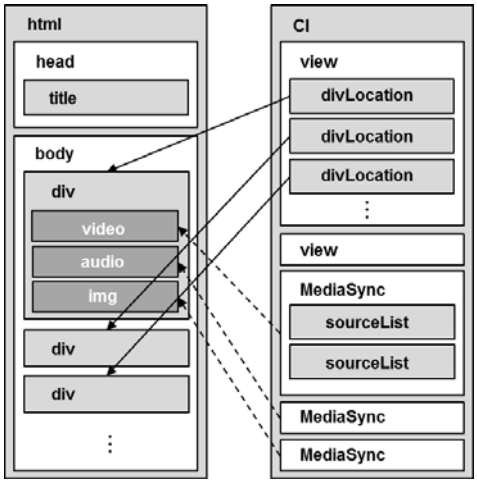
2.3 MMT-PI (MPEG Media Transport-Presentation Information)

MMT-PI is a document format that has spatial and temporal information of media used in MMT standard. MMT-PI is largely composed of HTML documents and MMT-CI (also known as MPEG CI) [6]. The HTML document conforms to the HTML5 format and contains spatial information of the initial state of the media.

The subsequent screen configuration information is made through the CI document. MMT-CI is a XML-formatted document that contains temporal information about the media content provided on the screen[6]. CI document references an HTML document to define elements within the HTML.

If a HTML5 document has spatial information of elements such as video, audio, etc., temporal information corresponding to the HTML5 document is configured in the CI. Figure 4 shows the structure of MMT-PI [6].

In this paper, MMT-PI is considered as a metafile that manages the entire MPD documents. The user can be informed about initial screen configuration of the service. When the user selects desired view location, MMT-CI and HTML document which is for the spatial information corresponding virtual location of the video service are



(Figure 4) Structure of MMT-PI

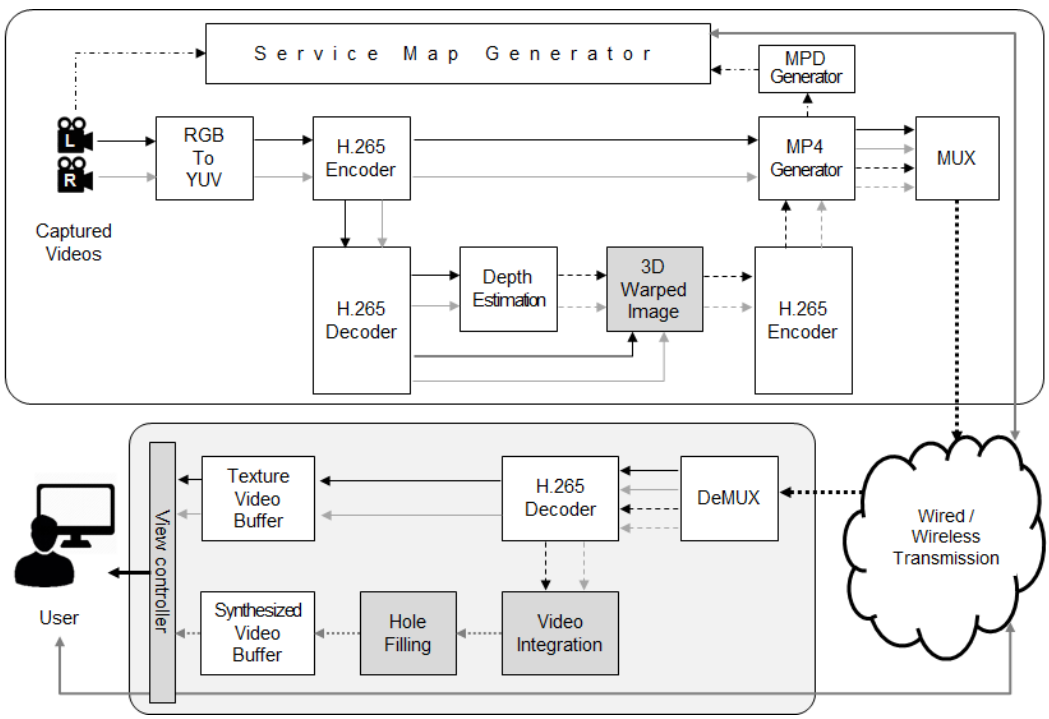
provided to the client side. In addition, MMT-CI includes internal information for synchronization between media data. This makes it easy to manage the relationship between every video and audio.

For the FVV service, video should be changed by the user's request and audio should be changed also according to the view change. In order to provide smooth FVV service, it is necessary to change audio and video naturally. Therefore, we use MMT-PI as FVV service map format considering these advantages in this paper.

3. Implementation Method for DASH-based Free-viewpoint Video Streaming System

3.1 Proposed FVV Streaming System

The proposed structure is shown in Figure 5. There may be a number of video files used for the entire service, but usually two video files are used for synthesizing views which referred to as left video and right video. The left and right sides are decided, when a virtual view is need to be synthesized in the middle of two video files. At the time of



(Figure 5) System Architecture of the Proposed Free-viewpoint Video Streaming Service

capturing, each camera is arranged by the distance calculation. The distance can be varied depending on the specifications of the camera or the overall stage size for the video service. Also it depends on how the service is provided. The overall service structure type can be dome, or cube, or a spherical structure. Information such as the number of cameras and the layout of the cameras are delivered to the video service map generator. Then, the captured video is converted and passed to the H.265 encoder. The encoded video files are encapsulated as MP4 box format for transmission.

The system structure has additional processes. In order to synthesize the video, it is performed with the left and right original video files, the left and right depth video files, and necessary information files for synthesis. Commonly used synthesis program performs 3D warping, video integration, and hole-filling with the above information. However, it takes about one second to generate a single frame of synthesized image when experimenting with the program, so it takes a considerable time for the user to receive the synthesized image. Therefore, it is necessary to synthesize faster in order to provide smooth service, so we designed a method to shorten the time which takes the most time in advance in the server side.

To do this, we suggest to prepare left and right original videos encoded as H.265 format. Then, we encapsulate the videos as MP4 box format, and we also sends the H.265 format video to the H.265 decoder. The reason is to prevent the problem that may occur on the client side, in order to reduce the difference between the image decoded by H.265 and the depth image generated in advance by the server and the 3D warped image. Then, a depth video is generated by depth estimation, and a 3D warped video is generated through the decoded image and the depth video. And then it is encapsulated as MP4 format. The server provides the original video and the 3D warped image at the same time, and the MPD document can be generated for each view with this information. The generated MPD document information is input into the service map generator, and the created service map is transmitted to the user. The user selects the view which is to be provided through the service map.

If the original video exists at the view point selected by the user, the corresponding video can be provided immediately. However, if the original video does not exist, the view can be synthesized through the left and right 3D warped images of the corresponding position.

3.2 Server Side

After the capturing of the video, the service configuration can be made after the number of the whole views, the distance between the defined videos, and the relationship information are delivered to the service map generator.

In addition, the media must be performed necessary tasks such as video conversion and encoding before configuring the service.

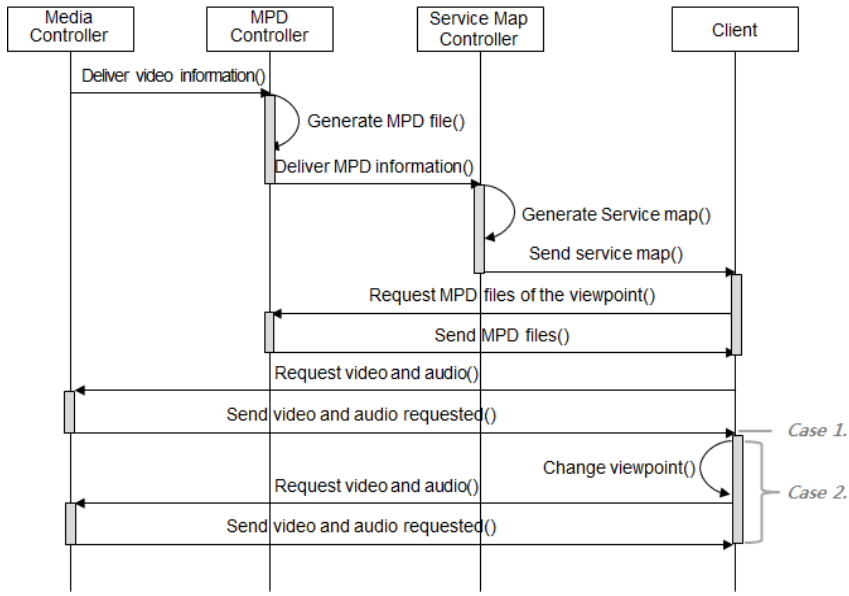
Figure 6 shows the work flow for server side systems. The media controller transfers the segment configuration information of the image to the MPD. The MPD controller configures the MPD document with the received information. Then, the MPD information is delivered to the service map controller so that each MPD can be mapped to the map by the viewpoint position. The generated service map is delivered to the client, through which the user can select the view of the service.

The response work for requests is simple compared to the initial workload of the server. If the video of the requested view position is close to or farther than the current viewpoint of the user, the operation of the server is different. In case 1, if the user requests a video of a view that does not have an original video which is near from, the server can to provide the original video and audio without any change. The spatial information of the service will not be changed, but the temporal information will be changing on the server.

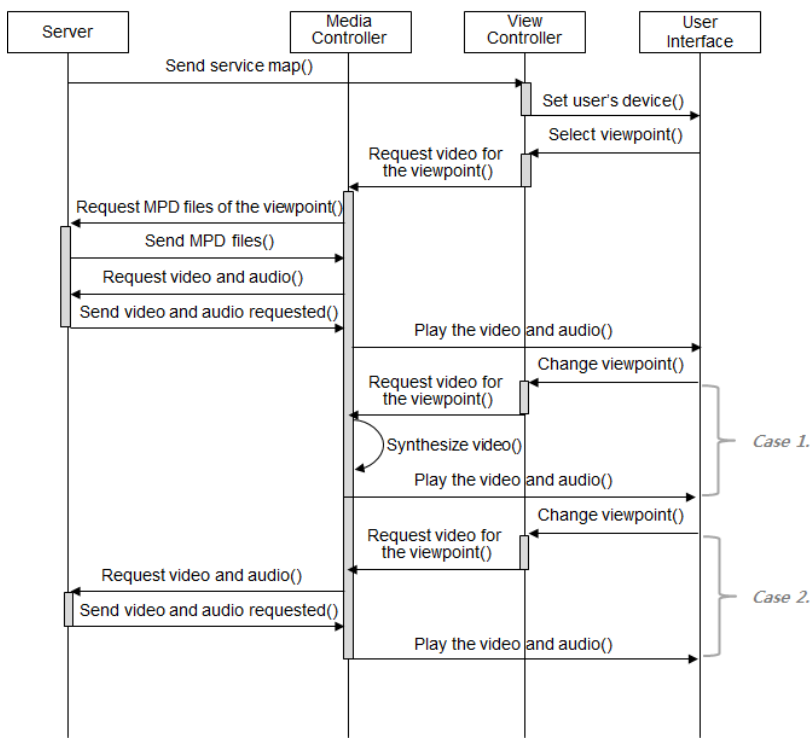
In case 2, when the user can select a new viewpoint image that is not expected, the client can access MPD of the corresponding location with service map information. the video and audio of the changed location in the MPD file are requested to the media controller in the server, and the server transmits the audio and video corresponding to the request.

3.3 Client Side

Figure 7 shows the work flow of the main functions within the client. It is most important to get a service map from the server to start the service. The service map contains information about how many views are provided and also the



(Figure 6) Sequence Diagram of Server System



(Figure 7) Sequence Diagram of Client System

relationship between each view. When the user selects a view point in the FVV service, the view controller requests the video first of the view from the media controller. The media controller downloads the file through the address of MPD that it has. At this time, other videos corresponding to the top and bottom or the left and right sides of the current video are downloaded.

This is because when the user switches the screen, it is more likely to select a view that is closer to the currently viewed video than the far-sighted view.

Also, moving to a nearer view rather than moving to a farther view should be smoother to improve the quality of the overall service. Therefore, it is assumed that the service downloads about 3 to 5 images depending on the network conditions. In the case of audio, the composition of the sound can be various depending on the view location. For example, we can suppose that a FVV service provides is an orchestra. When a musician is playing the violin on the left side of the orchestra and another musician is playing the cello on the right side, the violin's performance will be much louder on the left, and the cello's performance will be much louder on the right.

Depending on the service provider, there are different methods such as providing position control information related to one audio or controlling the sound by providing several audio. However, basically it is hard to adjust sounds in the service with few audio files smoothly, and it is impossible to get every audio file for each view location. So we assume that the service has audio and related information and adjust the audio to match the current video in the user's view. The media controller controls the video quality so that both video and audio are to be synchronized and the stable service can be provided.

Case 1 in figure 7 means that the user requests a view in which the original video at the position does not exist. If a user selects a view point in FVV service, the view controller requests video of the view to the media controller. The media controller downloads surrounding videos at the same time, and it can synthesize the video with the acquired videos. And it provides the synthesized video to the user.

Case 2 occurs when the user can select a video at view that is not expected. The user can select the view around the video that the user is watching, even if the original video does not exist. Because the surrounding video has already been downloaded, it can be used to synthesize viewpoints. However, if the user selects a remote view location, the service should provide only the view in which the original video exists when the view is initially changed.

The view was requested when the original video is not downloaded, so it is the best way to provide a stable service to the user by allowing the view to be accessible first and then changing to the surrounding point of view through interface control.

Once the user has requested a viewpoint change, the view controller makes a request to the media controller as usual. If the media controller requests a video that does not exist in the media controller, the media controller provides video and audio through the MPD file at that point on the server and plays it.

4. ConClusions

In this paper, we proposed an implementation method to make FVV streaming natural and smooth. The service map contains the necessary information for both providing and receiving services. The service provider can more easily manage the contents and can regularize the synchronization method or the screen switching method without individually setting it. The service map is provided at the beginning of the service. For stable service, the information such as video is included in each MPD, so that the video is downloaded and provided in real time.

Therefore, we generated a service map in the form of a PI document. Providing the service map as PI document format has two major advantages. First, we can define views that are available to users in the MPD, and make it easy to manage the entire service time and space information in conjunction with the CI. The synchronization problem is important to provide FVV services smoothly. Through this service map and documents, it is possible to consider the synchronization between video and audio, and between the converted video.

In addition, this method is expected to save time and cost in the client side. Our suggesting streaming system architecture which is more detailed than the existing suggestion. Also, we designed the system which can

synthesize easily on the client system.

Existing system performs synthesis process by on large module, but we divided it into several modules and put 3D warped video generation process to the server side to reduce time consuming. The process time take quite big importance in delay of the service, and it can drop the quality of the service. Moreover, we also presented the operation flow of server and client side. Through the proposed system, the server and the client can operate by following the proposed flow. We divided the user's view changes into two cases in this paper. The case 1 means that the user requests a view in which the original video does not exist, when the user and trying to change the view location. This can occur only in a position close to the previously viewed video.

When the user requests, there is no additional work on the server, because the original video and the 3D warped video are transmitted together from the beginning of the view. All the data is already transmitted that can be synthesized at the client side. However, additional work is required by the user's request on client side. Synthesized video is generated with surrounding original videos and 3D warped videos, and it is synthesized considering the flow of time.

The case 2 occurs when the user selects other view that is not anticipated. At this time, the client accesses the view controller and checks whether there is video of the corresponding viewpoint in the client. The media controller requests to the server through MPD of the video which is not downloaded yet. The server sends the requested video.

In this paper, we propose concrete module unit and operation flow by constructing system that utilizes service map actively. It will be possible to provide FVV service naturally and smoothly. It is clear that the network environment and the situation, and the processing capability of the terminal are very important factors in providing FVV streaming service. However, these problems do not improve in the short term, so it is important to find a way to provide the best service in the present situation.

Future research will focus on the operation of the user interface mapped with the service map. In order to provide natural service, it is important to minimize the delay time. There remains a problem such as what type of input method should be used to select view location information, how long the maximum delay time should be set which to be provided after the search, and what degree of sensitivity to react.

In addition, there is problem such as whether a separate audio file is generated and then synchronization is established, or additional information that can be modulated is provided and set when constructing audio information.

Changing the viewpoint also means that the user's virtual location is changed which is received for the service, so the immersiveness of the service degrades if the sound is the same. Finally, we will study focus processing on objects. Although the viewpoints are the same, they are not necessarily mapped to the same view. If the focus is on a certain object, the viewpoint can be fixed, but it is difficult to see a meaningful scene immediately when the position is changed with the simple position information without focus. Therefore, we will continue to conduct such research in the future.

참고문헌(Reference)

[1] M. Tanimoto, M. P. Tehrani, T. Fujii and T. Yendo, "Free-Viewpoint TV," IEEE Signal Processing Magazine, pp.67-76, 2011. https://doi.org/10.1109/MSP.2010.939077

[2] A. Hamza and M. Hefeeda, "A DASH-based Free Viewpoint Video Streaming System," In Proc. of the ACM Workshop on Network and Operating Systems Support for Digital Audio and Video, pp. 55-60, 2014. https://doi.org/ 10.1145/2597176.2578276

[3] A. Huszak, "Advanced Free Viewpoint Video Streaming Techniques," Multimedia Tools and Applications, January, pp.373-396, 2017. https://doi.org/10.1007/s11042-015-3048-9

[4] https://www.youtube.com/watch?v=lrMcdaEzonA, "Correspondence and Depth-Image Based Rendering".

[5] ISO/IEC 23009-1, "Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats," 2012.

[6] ISO/IEC 23008-1, "Information technology – High efficiency coding and media delivery in heterogeneous environments – Part1: MPEG media transport (MMT)", 2014.

[7] Y. Chen, Y. K. Wang, K. Ugur, M. M. Hannuksela, J. Lainema and M. Gabbouj, “The emerging MVC standard for 3D video services,” EURASIP Journal on Advances in Signal Processing, December, pp. 1-13, 2008.
https://doi.org/10.1155/2009/786015

[8] T. Su, A. Sobhani, A. Yassine, S. Shirmohommadi and A. Javadtalab, “A DASH Based HEVC Multi-View Video Streaming System,” Journal of Real-Time Image Processing, May, pp.1-25, 2015.
https://doi.org/10.1007/s11554-015-0504-8

[9] M. Domanski, M. Gotfryd, and K. Wegner, “View synthesis for multiview video transmission,” The 2009 International conference on image processing, computer vision, and pattern recognition, pp.1-4, 2009.

[10] A. Huszak, “Optimization of distributed free-viewpoint video synthesis,” 3DTV-Conference: The True Vision - Capture Transmission and Display of 3D Video, pp.1-4, 2014.
https://doi.org/10.1109/3DTV.2014.6874738

[11] Z. Han and Q. Dai, “A new scalable free viewpoint video streaming system over IP network,” IEEE international conference on acoustics speech and signal processing, pp.773-776, 2007.
https://doi.org/10.1109/ICASSP.2007.366350

[12] S. Jo, D. Lee, Y. Kim and C. Yoo, “Development of a simple free viewpoint video system,” Multimedia and Expo, 2008 IEEE International Conference on, June, 2008.
https://doi.org/10.1109/ICME.2008.4607750

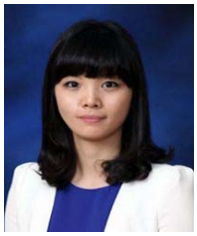
[13] R. Ma, T. Maugey and P. Frossard, “Optimized Data Representation for Interactive Multiview Navigation,” IEEE Transactions on Multimedia, December, 2017.
https://doi.org/10.1109/TMM.2017.2779039

[14] A. Smolic, “3D video and free viewpoint video－From capture to display,” Pattern Recognition vol. 44, September, 2010.
https://doi.org/10.1016/j.patcog.2010.09.005

[15] N. Hu, Y. Zhao and H. Bai, “3D View Synthesis with Feature-Based Warping,” KSII Transactions on Internet and Information Systems, vol. 11, no. 11, pp. 5506-5521, 2017.
https://doi.org/10.3837/tiis.2017.11.018

[16] L. Zhao, A. Wang, B. Zeng and J. Jin, “Scalable Coding of Depth Images with Synthesis-Guided Edge Detection,” KSII Transactions on Internet and Information Systems, vol. 9, no. 10, pp. 4108-4125, 2015.
http://dx.doi.org/10.3837/tiis.2015.10.019

◐ 저 자 소 개 ◑



서 민 재(Minjae Seo)
2013년 서울여자대학교 멀티미디어학과(공학사)
2016년 서울여자대학교 일반대학원 정보미디어학과(이학석사)
2016년～현재 서울여자대학교 대학원 컴퓨터학과 박사과정
관심분야 : 차세대 방송통신시스템, 차세대 영상서비스
E-mail : seominjae@swu.ac.kr

백 종 호(Jong-ho Paik)
1994년 중앙대학교 전기공학과(공학사)
1997년 중앙대학교 대학원 전기공학과(공학석사)
1997년～2011년 전자부품연구원 모바일단말연구센터 센터장
2007년 중앙대학교 대학원 전자전기공학부(공학박사)
2011년～현재 서울여자대학교 소프트웨어융합학과 부교수
관심분야 : 차세대 방송통신시스템, 차세대 영상시스템
E-mail : paikjh@swu.ac.kr