Implementation of WebRTC-based Hybrid Transmission System for 3D Visualization

Jungwook Wee, Minju Cho, and Younsung Lee
Contents Convergence Research Center
Korea Electronics Technology Institute
Seoul, Korea
jwwee@keti.re.kr, cmj1228@keti.re.kr, yslee@keti.re.kr

***Abstract*—In this paper, we propose and implement a WebRTC-based transmission system that can selectively utilize 3D object delivery and video streaming. The proposed method implemented based on the web has the advantage of being able to be serviced by various devices with browsers regardless of hardware performance. To evaluate the performance of the video stream latency for real-time 3D service, user interaction response time is measured through experiments. As a result, it is shown that the proposed method can provide real-time service.**

***Index Terms*—metaverse, 3D streaming, immersive, hybrid transmission**

I. Introduction

In recent years, many studies on 3D object rendering and transmission have been conducted due to the expansion of metaverse services [1-4]. A 3D object has variable complexity depending on its components and generally has a larger size than rendered 2D images. Since the rendering of complex 3D objects requires high-end hardware, real-time visualization is not possible on low-performance devices. To solve this problem, a 2D video streaming-based 3D visualization system has been proposed [3] [4]. Since these systems render 3D objects and transmit rendered images on the server, interaction latency inevitably exists. To reduce interaction latency, a method of streaming 3D content to a Dynamic Adaptive Streaming over HTTP (DASH) service is proposed [3]. DASH is an adaptive bitrate streaming technology that enables high-quality streaming of media content over the internet provided by conventional HTTP web servers [5]. In DASH, latency occurs according to the chunk size since the content is fragmented into and transmitted chunks. Although low-latency DASH has been proposed to reduce the latency, it has a minimum delay of 200 ms or more [6] [7].

In [4], a Real-Time Streaming Protocol (RTSP) based 3D visualization system has been proposed. This system is reduced latency by approximately 140 ms by streaming GPU-based zero-latency encoded images rendered on the server via the RTSP. However, this method requires dedicated software because it consists of separate protocols for video streaming and interaction message processing.

This research was supported by Institute of Information Communications Technology Planning Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2021-0-00762, Development of viewpoint-based XR visualization and 3D automatic transformation technology for large-scale CAD)

In this paper, we propose and implement a web Real-Time Communication (WebRTC) based hybrid transmission system that can selectively utilize 3D object transmission and video streaming for 3D visualization. The proposed method enables low-latency 3D content service in the web browser regardless of the device's performance.

II. Web Real-Time Communication

Web Real-Time Communication (WebRTC) is an open-source and free project that aims to embed real-time voice, data, video, and instant messaging capabilities through Javascript APIs within Web browsers [8]. Peer-to-peer (P2P) connection to reduce transmission latency is made through Signaling, Interactive Connectivity Establishment (ICE), Session Traversal Utilities for NAT (STUN), and Traversal Using Relays around NAT (TURN) servers. When using Network Address Translation (NAT), the P2P connection is established after finding the user address through the STUN and TURN servers and exchanging user information using the ICE servers. Data is streamed using Secure Real-Time Transport Protocol (SRTP) and Stream Control Transmission Protocol (SCTP) when the P2P connection is established. Fig. 1 and Fig. 2 show the P2P connection structure and network protocol stack of WebRTC, respectively.

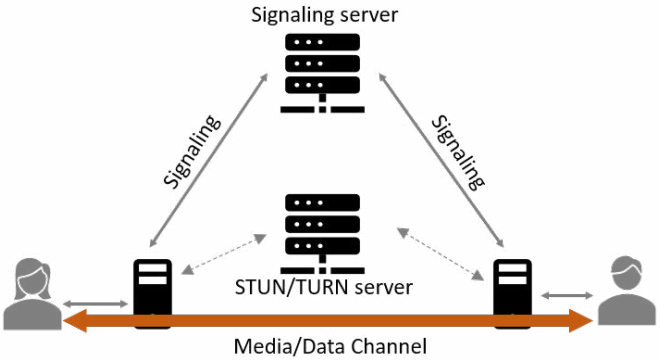

Fig. 1. Structure of P2P connection

III. WebRTC-based Hybrid Transmission System

In traditional 3D services mainly used in the game, the user device downloads and render all 3D objects. The object

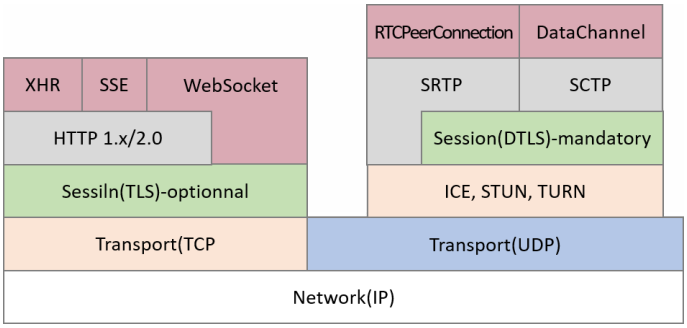

Fig. 2. Structure of P2P connection

download method causes an initial download delay, but there is no latency for interaction. However, high-quality 3D objects similar to the real world have the disadvantage of taking a long time to download and requiring high-performance hardware for rendering. The streaming-based method, which has a low initial delay, has a latency for user interaction, making it difficult to provide real-time services. In this section, we propose and implement a WebRTC-based hybrid transmission system based that can overcome the disadvantages of each method.

In the case of the object downloading method, the user device downloads, renders, and controls the 3D objects files using REST API. In the case of the streaming method, the device receives object control information via the data channel and rendered image streams via the media channel. A user ID is assigned to one channel to ensure a 1:1 connection between the streaming server and the user. To reduce graphic processing delay due to multi-object rendering by multi-user requests, Web Worker-based Offscreen Canvas (virtual canvas) [9] [10] is applied to increase rendering speed by directly generating stream data without drawing on the screen. The virtual canvas was set to the same size as the resolution of the user device to optimize the use of resources on the server side. Fig. 3 shows the proposed system structure, and the main functions are as follows.

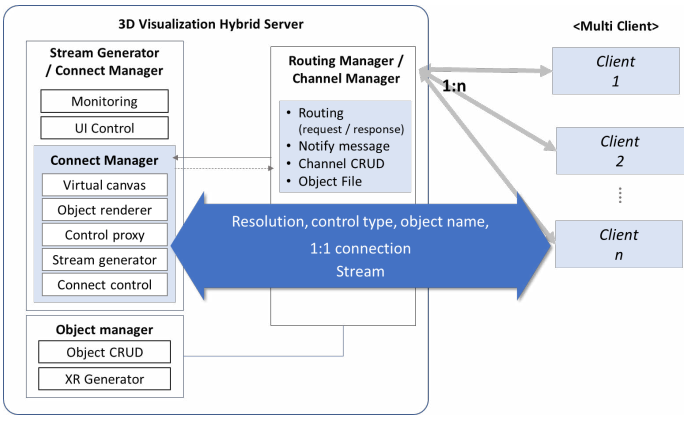

Fig. 3. Structure of the proposed system

- *Routing Manager* - When the user accesses the service page, the contents server delivers the path of *Channel Manager* and 3D service list, and registers the user.
- *Channel Manager* - Create or delete a new channel for *Connect Manager* when the user logs in or out of the web service. The *Channel Manager* performs as the signaling server that exchanges the user's location on the network with *Connect Manager* for the P2P connection. It also manages the object name, user address, *Connect Manager* address, channel name, and service type (downloading or streaming).
- *Connect manager* - Performs P2P connection with the user using the unique socket address allocated from the *Channel Manager*.
- *Stream Generator* - Renders 3D objects, encodes rendered images, and creates media streams using virtual canvas, as shown in Fig. 4. The created media stream is streamed to the user through the *Connect Manager*. The proposed hybrid transmission server can provide 3D streams to multiple users simultaneously by creating a *Stream Generator* for each user.

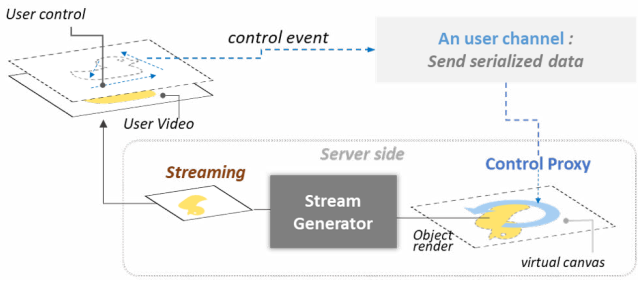

Fig. 4. The process of the stream generator

The process of a user requesting and receiving 3D content services through the hybrid server is shown in Fig. 5 and is as follows.

- When a client accesses the service page, the user receives the URL and 3D object content list from the *Connect Manager* after being authenticated by the *Routing Manager*. *Channel Manager* allocates a channel for *Connect Manager* for peer connections with users and registers the user ID as the channel name for channel management.
- When the user obtains the network location of the *Connect Manager* through the *Channel Manager* and connects to the *Connect Manager*, streams and data channels are created.
- In the case of the 3D object downloading service, the user ID and requested contents are delivered to the *Channel Manager*, and *Channel Manager* transmits 3D object files to the user through *Routing Manager*. After downloading the 3D object, rendering and controlling can be operated on the user's devices.
- In the case of the streaming service, user ID, object name, device resolution, etc. are delivered to the *Channel Manager*. The *Channel Manager* updates the user's

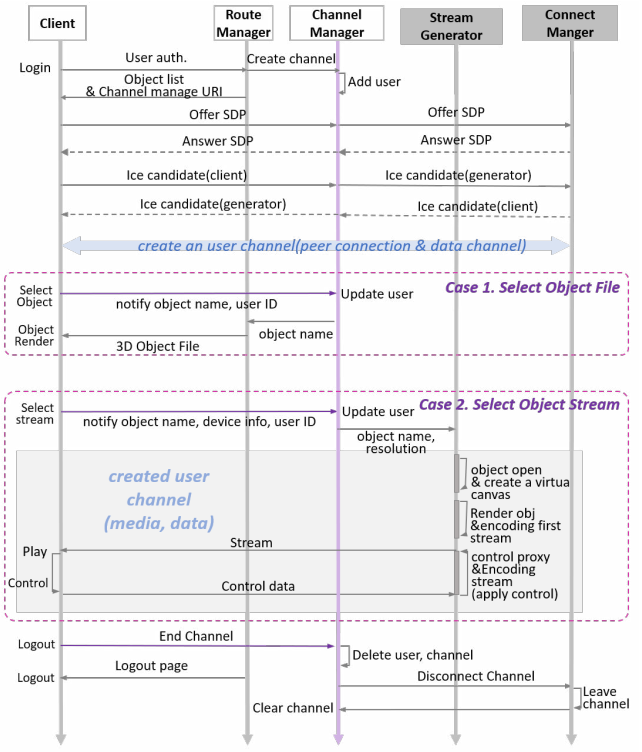

Fig. 5. The process of the stream generator

request information and passes the object information to the *Stream Generator*.

- The *Stream Generator* streams rendered images of 3D objects to the user at the requested resolution. User-controlled events are passed to the *Stream Generator* via the data channel in the *Connect Manager*. *Stream Generator* transforms 3D objects depending on the control information, renders transformed objects, and streams rendered images. This process is repeated while the user controls the contents.
- When the user exits the service or leaves the page, the *Channel Manager* removes the user from the list and terminates the P2P connection. The user channel is deleted and the user is logged out by the *Routing Manager* when the *Connect Manager* leaves the user channel.

IV. Expermental Results

The proposed system is implemented by nodeJS, Javascript, three.js, and Electron, and the STUN server has used a server provided by Google. The user devices used for verification are a PC, a tablet, and a mobile device with Chrome browsers. Fig. 6 shows the result of the implemented system. In the figure, the left and right represent the monitoring screen of the server including the stream generator and user devices, respectively. As shown in the figure, it can be known that the service is available on various devices.

To evaluate latency performance, we performed measurement experiments on motion-to-photon (MTP) latency [4]. MTP latency was measured 50 times for the experiment and Table I shows the result of the comparison for the previous and the proposed method. Although the proposed method based on WebRTC has a latency of approximately 20 ms compared to the standalone method of [4], it was confirmed that there is no problem with real-time visualization.

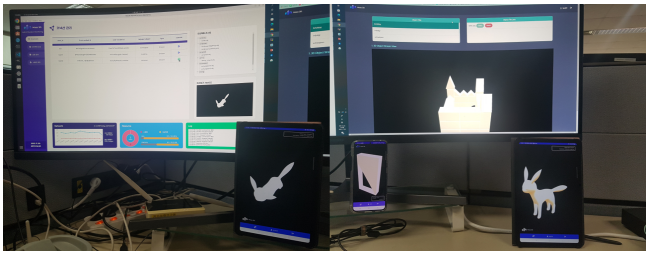

Fig. 6. The process of the stream generator

TABLE I
Results of Latency Performance

	Standalone	Proposed
Resolution	FHD	FHD
Frame rate	30	30
Codec	HEVC	H264
Latency	140.54 ms	160.37 ms

V. Conclusion

In this paper, we proposed and implemented a WebRTC-based hybrid streaming system. By applying the web-based technology, it was confirmed that the service is available on various devices with browsers. Moreover, it was possible to provide a low-latency service even when streaming 3D objects by creating a user-only channel with WebRTC.

References

[1] Y. Kim, Y. Choi, Y. Lee, H. Han, and S. Kang, "E-Render: Enabling UHD-Quality Cloud Gaming Through Edge Rendering," in IEEE Access, Vol 10, pp. 72107-72119, 2022.
[2] S. Gul, D. Podborski, T. Buchholz, T. Schierl, and C Hellge, "Low-latency Cloud-based Volumetric Video Streaming Using Head Motion Prediction," in 30th ACM Workshop on NOSSDAV, 2020.
[3] S. Song, M. Cho, and J.Wee., "Design of DASH based Hybrid Transmission System for Immersive Content," The 13th International Conference on Internet (ICONI), pp. 351-352, Dec. 2021.
[4] J. Wee, M. Cho, "An Implementation of Low-Latency 2D Streaming based 3D Visualization Systems," The 14th International Conference on Internet (ICONI), pp.21-22, Dec. 2022.
[5] I. Sodagar, "The MPEG-DASH standard for multimedia streaming over the Internet", IEEE Multimedia Mag., vol. 18, no. 4, pp. 62-67, Apr. 2011.
[6] L.Sun.T.Zong, S.Wang, Y.Liu, Y.Wang, "Towards Optimal Low-Latency Live Video Streaming" IEEE/ACM Transactions on Networking, Volume: 29, Issue: 5, pp.2327-2338, Oct. 2021.
[7] N. Bouzakaria, C. Concolato, and J. Le Feuvre, "Overhead and performance of low latency live streaming using MPEG-DASH", Proc. 5th Int. Conf. Inf. Intell. Syst. Appl. (IISA), pp. 92-97, Jul. 2014
[8] https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API/
[9] https://developer.mozilla.org/ko/docs/Web/API/Web_Workers_API/Using_web_worker
[10] https://developer.mozilla.org/en-US/docs/Web/API/OffscreenCanvas