International Journal of Internet, Broadcasting and Communication Vol.14 No.3 276-284 (2022)
http://dx.doi.org/10.7236/IJIBC.2022.14.3.276

IJIBC 22-3-32

Performance Analysis of Cloud Rendering Based on Web Real-Time Communication

Gyubeom Lim[1], Sukjun Hong[2], Seunghyun Lee[3], Soonchul Kwon*[4]

[1]M.S, Department of Plasma Bio Display, Kwangwoon University,Seoul, South Korea;
[2]M.S, Department of Smart System, Kwangwoon University,Seoul, South Korea;
[3]Professor, Ingenium College Liberal Arts, Kwangwoon University, Seoul, South Korea;
*[*4]Associate professor, Graduate School of Smart Convergence, Kwangwoon University, Seoul, South Korea;*

*dlarbqja97@kw.ac.kr, 000jun2@kw.ac.kr, shlee@kw.ac.kr, *ksc0226@kw.ac.kr*

Abstract

In this paper, we implemented cloud rendering using WebRTC for high-quality AR and VR services. Cloud rendering is an applied technology of cloud computing. It efficiently handles the rendering of large volumes of 3D content. The conventional VR and AR service is a method of downloading 3D content. The download time is delayed as the 3D content capacity increases. Cloud rendering is a streaming method according to the user's point of view. Therefore, stable service is possible regardless of the 3D content capacity. In this paper, we implemented cloud rendering using WebRTC and analyzed its performance. We compared latency of 100MB, 300MB, and 500MB 3D AR content in 100Mbps and 300Mbps internet environments. As a result of the analysis, cloud rendering showed stable latency regardless of data volume. On the other hand, the conventional method showed an increase in latency as the data volume increased. The results of this paper quantitatively evaluate the stability of cloud rendering. This is expected to contribute to high-quality VR and AR services

Keywords: *Augmented Reality, Cloud Rendering, Rendering, Streaming, Web, WebRTC*

1. Introduction

Recently, mobile device distribution and hardware and graphics processing technologies have advanced. With the advancement of technology, the accessibility of the web has increased. The web has evolved from a simple static text and image delivery platform to a platform containing dynamic content [1]. With the development of the web and improved accessibility, large-capacity content such as videos, games, VR, and AR can be used [2]. The Conventional method downloads content when accessing the web. This method increases the latency as the capacity of the content increases. Latency is a factor that can substantially affect the user experience in a mobile network. As the latency of the user increases, the satisfaction of the user decreases. If the user's satisfaction is not satisfied by a certain level or more, there is a problem that the user's departure occurs. [3,4] Therefore, in this paper, we implement cloud rendering to reduce latency compared to conventional methods. Cloud Rendering is a streaming method based on the user's point of view. Cloud

Manuscript Received: July. 28, 2022 / Revised: August. 2, 2022 / Accepted: August. 5, 2022
Corresponding Author: kcs0226@kw.ac.kr
Tel: +82-2-940-8637, Fax: +82-50-4174-3258
Associate professor Graduate School of Smart Convergence, Kwangwoon University, South Korea

Rendering consists of Node.js server and Unity that provides streaming screen. Cloud Rendering uses WebRTC to implement real-time communication. As an evaluation method, the conventional web download method and the streaming method of cloud rendering are compared. The cloud rendering performance test measured FPS. As a measurement method, we measure the content latency of Cloud Rendering and the traditional method with a 3D model of the same capacity in the same Internet environment. Cloud Rendering's FPS measurement is measured using 3D AR content of 100, 300, and 500 MB.

2. Background Theories

2.1 WebRTC

WebRTC is a real-time communication technology through a web browser or JS API (JavaScript Application Programming Interface) without installing plug-ins or applications. WebRTC uses a P2P (Peer-to-Peer network) connection. [5-7] In the P2P method, each peer agrees to P2P communication. Thereafter, the connection between the Peers is made through the server. Connected Peers communicate by exchanging media data. Peers' data exchange can be found in Figure 1.

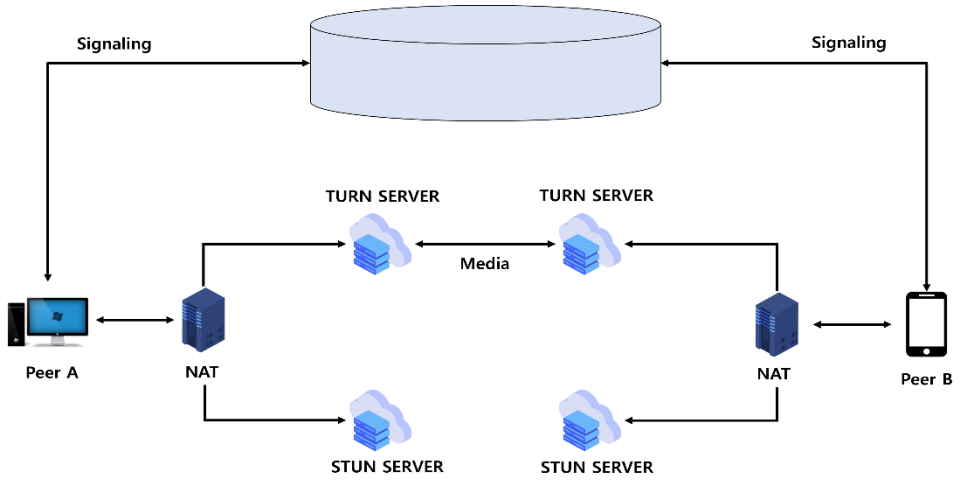


Figure 1. Web RTC

2.2 Cloud Computing

Cloud Computing is computing that provides IT resources as a service by utilizing Internet technology. Cloud computing is computing that borrows and uses IT resources (software, storage, server, network) as needed. It pays according to the service load generated and supports real-time scalability [8]. Cloud computing has the advantage of borrowing IT resources and using them at low cost. However, there are disadvantages such as corporate technology and service monopoly, dependence, and security problems [9]. Components of cloud computing include server scalability, system availability, and data reliability. It scales the virtual machine flexibly according to the user's load and stores data centrally. Because data is stored centrally, sharing and collaborating on one data is possible [10].

3. Proposed Method

In this paper, we implement cloud rendering using WebRTC. It makes 3D content available in a streaming method. Cloud Rendering communicates over an internal network in a P2P manner. IP address sharing between peers and port connections are made through STUN (Session Traversal Utilities for NAT) [11] servers. Internal network communication is via a TURN (Traversal Using Relays around NAT) [12]

server. Cloud Rendering's communication using WebRTC can be found in Figure 2.

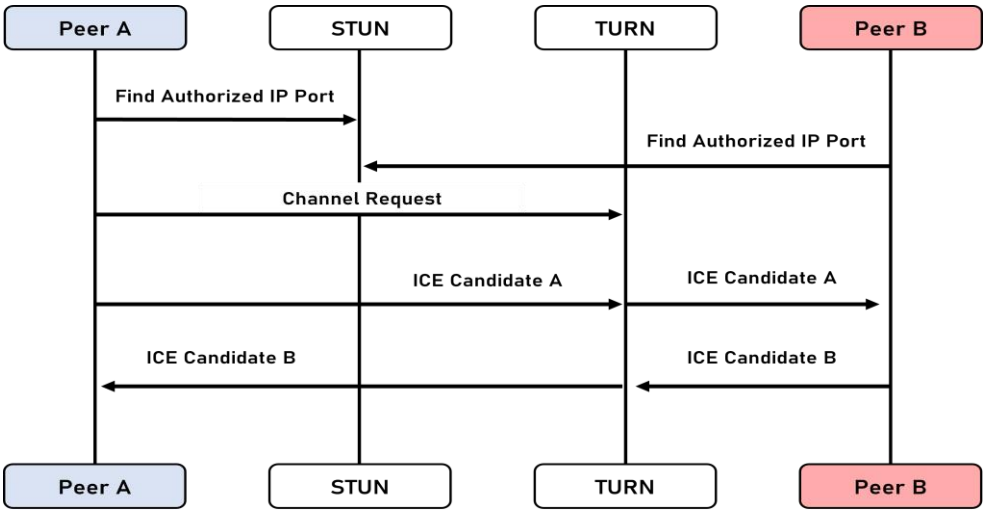


Figure 2. WEB RTC Protocol

3.1 Cloud Rendering

Cloud Rendering consists of Peers that provide content and Peers that use it. Peers communicate in real time. Peer, which provides content, was implemented using game engine Unity. The server for the connection between Peers was implemented as Node.js. Node.js is the role of a TURN server, enabling the connection between Peers and the transmission and reception of all data.

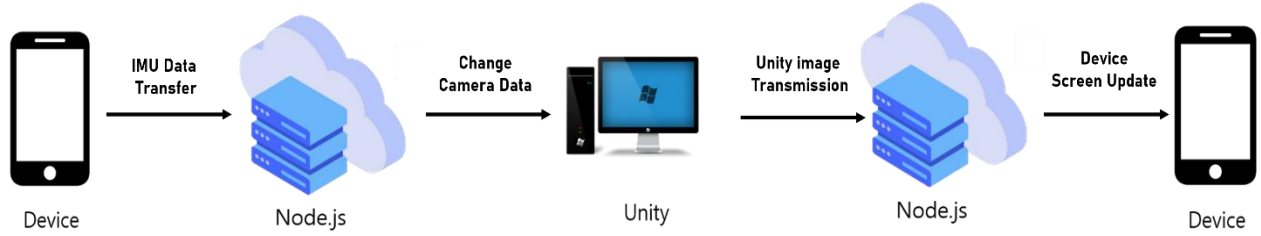


Figure 3. Cloud Rendering Signaling

Signaling of Cloud Rendering can be seen in Figure 3. Signaling process transmits data generated from mobile devices to Node.js. The transmitted data is sent and converted to Unity. Image conversion data created from unity is send to Node.js. Unity data updates through data transfer from Node.js to mobile devices.

Through the signaling process, the data generated from the mobile device is processed by the peer that provides the content. Since data processing is performed by the peer to provides the content, download does not occur when using web content.

4. Experiments

4.1 Experimental Environment

Table 1. Experimental Enviroments

	Galaxy S22 Ultra	Computer
CPU	Octa-Core	Intel Core i7-10700K - 3.8GHz
GPU	Snapdragon 8 Gen1	NVIDIA GeForce RTX 2060
RAM	12GB	32GB
OS	Android	Windows 10

Table 1 shows the device performance used in the experiment. The latency of the conventional method and Cloud Rendering was compared when using 3D content on a mobile device. The Conventional method is a method in which a mobile device downloads 3D content when accessing the web. Cloud Rendering provides 3D content through a server when a mobile device accesses the web.

4.2 Experiments Method

In this paper, we compare and evaluate cloud rendering with conventional methods of WebAR. WebAR and Cloud Rendering measure and compare under the same conditions. Cloud Rendering's evaluation measures FPS in different capacities of 3D AR content and 300Mbps of Internet environment.

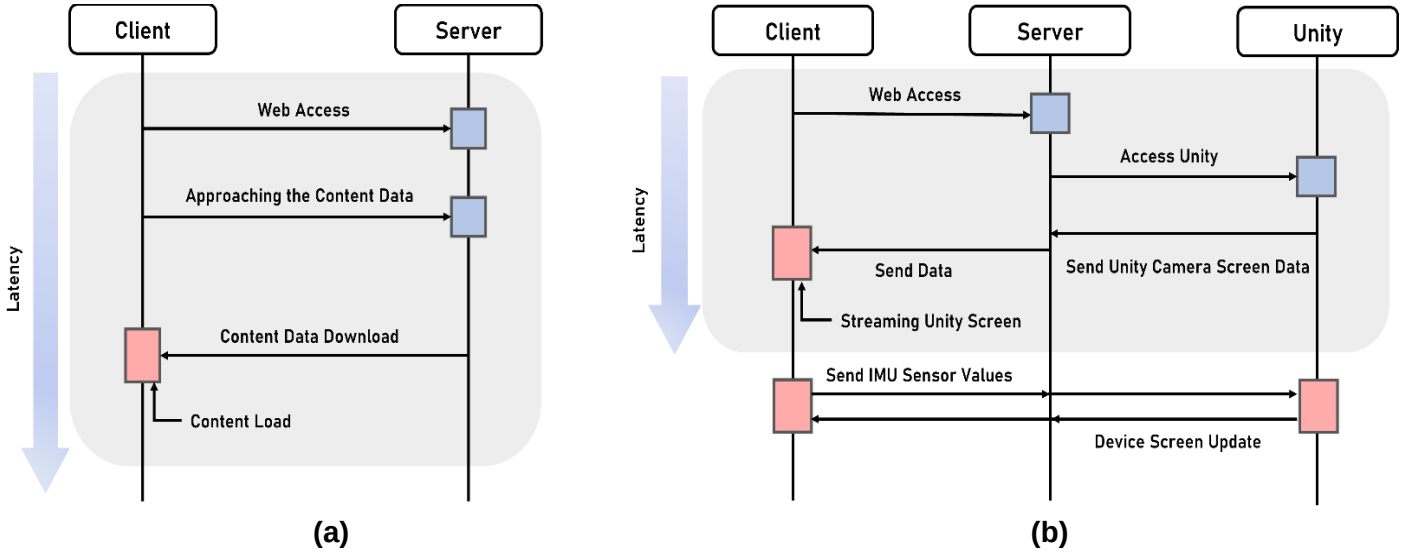


Figure 4. Content Latency
(a) Web AR (b) Cloud Rendering

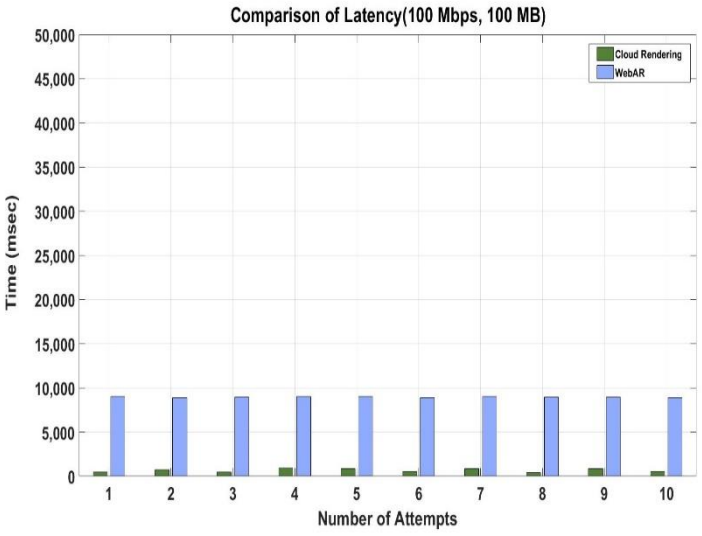
Figure 4 shows the structure and measurement method of WebAR and Cloud Rendering. Content loading times were measured down to Latency. Figure 4 (a) shows the structure of the web AR. It downloads the model when it accesses the web. When the download is complete, the model is rendered on the screen of the mobile device. Figure 4 (b) shows the structure of Cloud Rendering. It sends data from the server to Unity when accessing the web. Once the connection is confirmed, the data sent by Unity is rendered on the screen of the mobile device. Latency is measured until the time the model is rendered.

The FPS of cloud rendering measures the continuous data exchange after the figure's latency. The measurement is performed with a model with different capacities, and the performance difference according to the size of the model is checked.

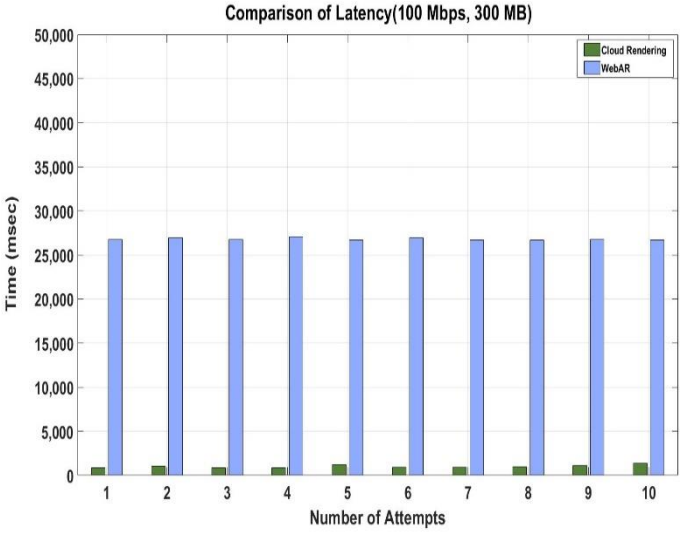
5. Experimental Result

5.1 Cloud Rendering Latency Comparison with Download Method

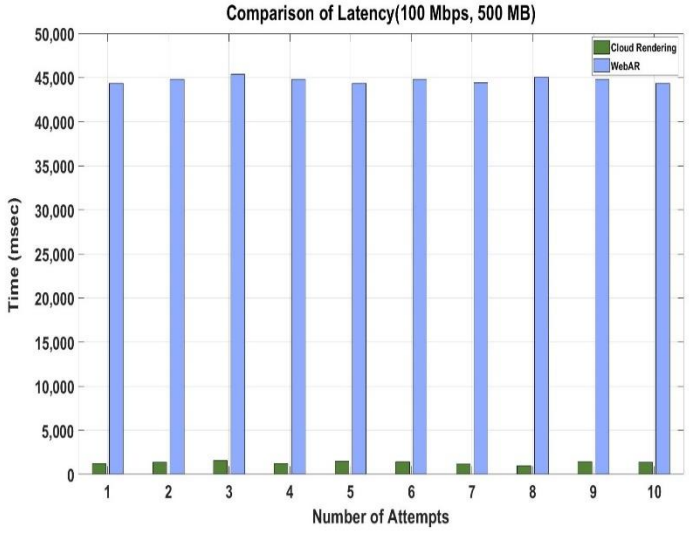
WebAR and Cloud Rendering evaluated web service latency with 3D AR content of 100 MB, 300 MB, and 500 MB in 100 Mbps and 300 Mbps Internet environments. Figure 5 is a graph measuring WebAR and Cloud Rendering 10 times.



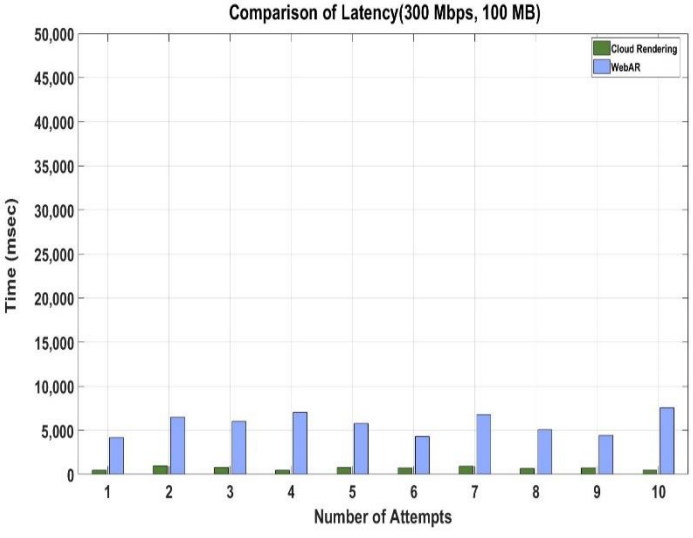
(a)



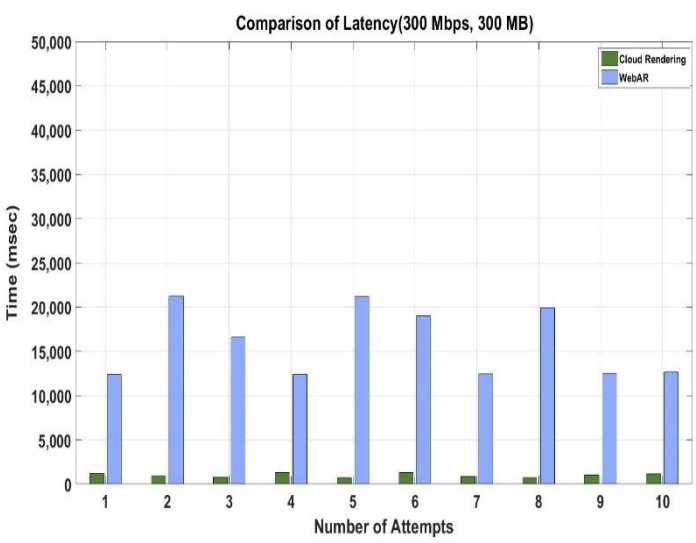
(b)



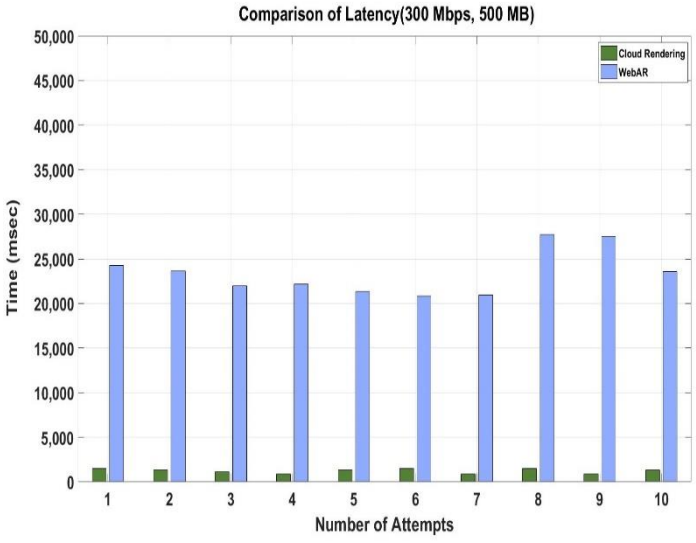
(c)



(d)



(e)



(f)

Figure 5. Comparison of Latency
(a) 100 Mbps, 100 MB Model (b) 100 Mbps, 300 MB Model (c) 100 Mbps, 500 MB Model (d) 300 Mbps, 100 MB Model (e) 300 Mbps, 300 MB Model (f) 300 Mbps, 500 MB Model

Table 2. Download Method Latency

Speed (Mbps)	Model Capacity (MB)	Maximum Latency (ms)	Minimum Latency (ms)	Average Latency (ms)	Standard Deviation (ms)
100	100	9,037	8,916	8,970	47.49
	300	27,059	26,703	26,801	131.77
	500	45,378	44,343	44,716	325.41
300	100	7,530	4,196	5,762	1,156.01
	300	21,271	12,393	16,054	3,760.72
	500	27,714	20,803	23,393	2,391.21

Table 3. Cloud Rendering Latency

Speed (Mbps)	Model Capacity (MB)	Maximum Latency (ms)	Minimum Latency (ms)	Average Latency (ms)	Standard Deviation (ms)
100	100	936	416	682	191.57
	300	1,342	841	1,013	164.06
	500	1,588	982	1,333	166.01
300	100	959	450	711	174.87
	300	1,309	732	1,009	223.07
	500	1,494	850	1,204	256.03

Table 2 and Table 3 show the detailed figures in Figure 5. In the case of WebAR, the average delay time for downloading to the client was 8,970 ms, 26,801 ms, 44,716 ms at 100 Mbps, and 5,762 ms, 16,054 ms, and 23,393 ms at 300 Mbps. When the model capacity increases by 100 MB, it is confirmed that the average

delay time increases by about 8,925 ms at 100 Mbps and by about 4,476 ms at 300 Mbps. In WebAR, when the model capacity increases by 100 MB, the average delay time increases by about 99% compared to 300 Mbps for 100 Mbps. In the case of WebAR, it was confirmed that it was greatly affected by the Internet speed and the capacity of the model. In the case of Cloud Rendering, the average delay time for downloading to the client was 682 ms, 1,013 ms, 1,333 ms at 100 Mbps, and 711 ms, 1,009 ms, and 1,204 ms at 300 Mbps. In Cloud Rendering, when the model capacity increased by 100 MB, the average latency increase was 164ms at 100 Mbps and 197 ms at 300 Mbps, which decreased about 17% from 100 Mbps.

As a result of comparing Table 2 and Table 3, Cloud Rendering showed better results for WebAR. Also, the average standard deviation of WebAR was about 167 ms at 100 Mbps and about 2,435 ms at 300 Mbps, showing a big difference. The average standard deviation of Cloud Rendering was about 173 ms at 100 Mbps and 217 ms at 300 Mbps, showing stable performance compared to WebAR.

Through the results, it was confirmed that Cloud Rendering is more stable than WebAR. Accordingly, it was confirmed that the streaming method of cloud rendering was more stable and the content delay time was reduced than the conventional web download method.

5.2 Cloud Rendering Performance Evaluation

Cloud Rendering's performance evaluation measured FPS with 3D AR content of 100 MB, 300 MB, and 500 MB in an Internet environment of 300 Mbps.

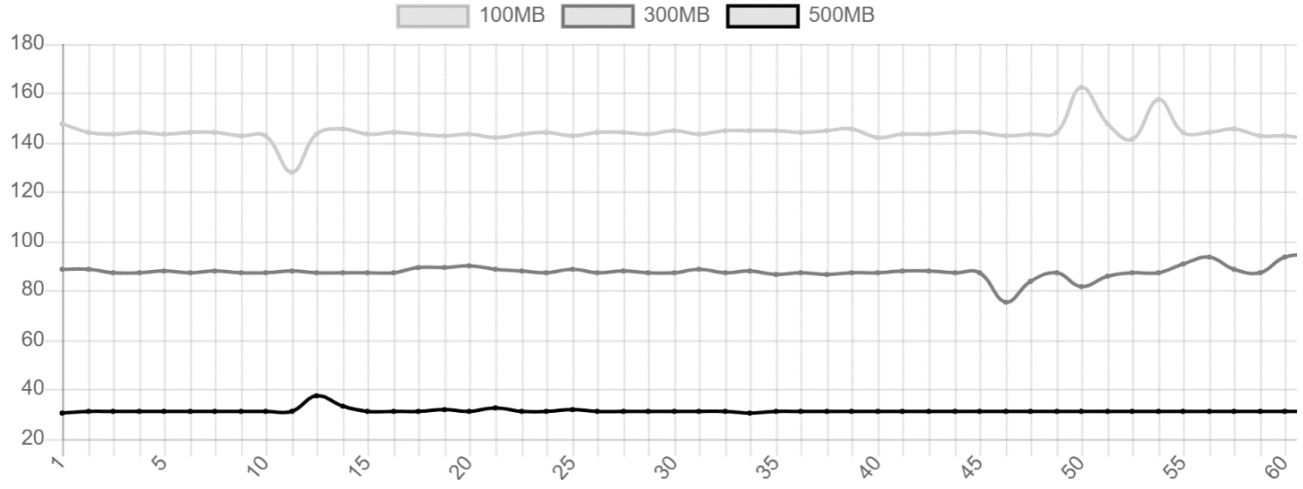


Figure 6. Cloud Rendering Performance

Table 4. Performance Comparison of each Model Capacity in Cloud Rendering

Model Capacity (MB)	FPS
100	144.06
300	87.70
500	31.40

Figure 6 shows the FPS of Cloud Rendering. If you look at this, you can see stable results. However, differences in performance according to content capacity can be confirmed. It can be seen that the performance decreases as the capacity of the content increases. Table 4 summarizes the FPS of Cloud Rendering. It can be seen that the FPS of Cloud Rendering is 144.06 FPS, 87.7 FPS, and 31.4 FPS,

respectively. Latency of Cloud Rendering is not significantly affected by the capacity of the content, but it was confirmed that it is significantly affected in the case of FPS.

6. Conclusion

In this paper, we implement Cloud Rendering to reduce latency when using high-capacity content such as AR and VR on the web. The Conventional method was greatly influenced by the Internet environment and content capacity. It has a problem in that the larger the capacity of the content, the longer the latency.

Cloud Rendering was implemented in a streaming method using WebRTC. Streaming methods are not significantly affected by the Internet environment and the capacity of content. Therefore, it is possible to reduce the latency when using the content. Through this, mobile latency is expected to be shortened when using large amounts of content. However, in the performance evaluation of Cloud Rendering, it was confirmed that FPS is affected by the capacity of the content. As a result of the experiment, real-time performance was confirmed in AR contents of 100 MB and 300 MB. The 500 MB model confirmed the performance that is difficult to provide real-time services. If cloud rendering is optimized in the future, it is expected that more than 500 MB of content will be available in real time. In addition, Cloud Rendering processes data on other devices, so the higher the performance of the device providing content, the better the results are expected on the same mobile device.

In the future research direction, research is needed on how Cloud Rendering reduces user latency when using high-end, high-capacity games, and content as well as web content.

Acknowledgement

The present research has been conducted by the excellent researcher support project of Kwangwoon University in 2022

References

[1] Yu, Hye-Rim, and In-Guk Song, *Evolution of social network service according to web service type change*, University of dankook, Gyeonggi-do, 2010

[2] Kim Cheol-Ki, “Trends Research and Development View of Mobile Augmented Reality Based on Smart Phone,” Journal of Korea Design Forum, no. 27, pp. 53–64, May 2010. DOI: https://doi.org/10.21326/ksdt.2010..27.005

[3] Kwon, Ohkyun et al., “Meaning of Waiting Experience and Principles of Service Design,” The Journal of the Korea Contents Association, vol. 17, no. 1, pp. 270–286, Jan 2017. DOI: https://doi.org/10.5392/JKCA.2017.17.01.270

[4] H. Kim, J. Park, M.-H. Choi, and I. Moon, “Web Content Loading Speed Enhancement Method using Service Walker-based Caching System,” Journal of Advanced Navigation Technology, vol. 23, no. 1, pp. 55–60, Feb 2019. DOI: https://doi.org/10.12673/jant.2019.23.1.55

[5] Getting started with WebRTC. *https://webrtc.org/getting-started/overview*

[6] Bak Se Chan, Im Geon Hyeong, Shin Hyeon Seok, Hwangbo Jun Su, Lee You Rak, *Technology trend analysis on Web-RTC*, University of Kyungpook, Daegu, 2020

[7] B. Sredojev, D. Samardzija, and D. Posarac, “WebRTC technology overview and signaling solution design and implementation,” 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). IEEE, May 2015. DOI: https://doi.org/10.1109/MIPRO.2015.7160422

[8] Lee, K.C. and Lee, Seung-Yun, “The Trends and Strategy of Standardization on Cloud Computing,” Electronics and Telecommunications Trends, vol. 25, no. 1, p. 90, Feb 2010. DOI: https://doi.org/10.22648/ETRI.2010.J.250109

[9] L. Jihoon, K. Eunchoong, B. Kiyoung, L. Yujin and K. Yong "An Application Method Study on the Electronic

Records Management Systems based on Cloud Computing" Journal of Korean Society of Archives and Records Management, vol. 14, no. 3, pp. 153–179, Aug 2014. DOI: https://doi.org/10.14404/JKSARM.2014.14.3.153
[10] D. S. Linthicum, “Cloud Computing Changes Data Integration Forever: What’s Needed Right Now,” IEEE Cloud Computing, vol. 4, no. 3. Institute of Electrical and Electronics Engineers (IEEE), pp. 50–53, 2017.
DOI: https://doi.org/10.1109/MCC.2017.47
[11] Session Traversal Utilities for NAT (STUN). *https://datatracker.ietf.org/doc/html/rfc5389*
[12] Traversal Using Relays around NAT (TURN). *https://datatracker.ietf.org/doc/html/rfc5766*