Article

Measurement Study of Real-Time Virtual Reality Contents Streaming over IEEE 802.11ac Wireless Links

Gusang Lee [1], Won Joon Yun [1], Yoo Jeong Ha [1], Soyi Jung [1,]*, Jiyeon Kim [2], Sunghoon Hong [2], Joongheon Kim [1] and Youn Kyu Lee [3,]*

1 School of Electrical Engineering, Korea University, Seoul 02841, Korea; rntkd0917@korea.ac.kr (G.L.); ywjoon95@korea.ac.kr (W.J.Y.); annaha17@korea.ac.kr (Y.J.H.); joongheon@korea.ac.kr (J.K.);
2 Samsung Advanced Institute of Technology, Suwon 16678, Korea; jiyeon31.kim@samsung.com (J.K.); ar.sung.hong@samsung.com (S.H.)
3 Department of Information Security, Seoul Women's University, Seoul 01797, Korea
* Correspondence: jungsoyi@korea.ac.kr (S.J.); younkyul@swu.ac.kr (Y.K.L.); Tel.: +82-2-3290-3223 (S.J.); +82-2-970-5601 (Y.K.L.)

check for updates

Citation: Lee, G.; Yun, W.J.; Ha, Y.J.; Jung, S.; Kim, J.; Hong, S.; Kim, J.; Lee, Y.K. Measurement Study of Real-Time Virtual Reality Contents Streaming over IEEE 802.11ac Wireless Links. *Electronics* **2021**, *10*, 1967. https://doi.org/10.3390/electronics10161967

Academic Editor: Imran Shafique Ansari

Received: 15 July 2021
Accepted: 14 August 2021
Published: 15 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

Abstract: Experience sharing among multiple users in virtual reality (VR) is one of the key applications in next generation wireless systems. In this VR application, one object can be reproduced as a virtual object based on recorded/captured multiple real-time images from multiple observation points. At this time, VR applications require a lot of bandwidth to provide seamless services to users in wireless links, and thus, a certain level of data rates should be maintained. As the number of users increases, the server allocates more data rates to users on top of the limited bandwidth in wireless networks. At this time, users who utilize the VR streaming services will suffer from a lower quality, due to the limited bandwidth. This paper reports the measurement study and also analyzes the fluctuations in terms of the data rates as the number of users increases while sharing point cloud information in real-time authorized reality environments over IEEE 802.11ac wireless networks. Moreover, it measures and analyzes fluctuations in terms of frames-per-second and Jitters, which are practical quality reduction indicators.

Keywords: virtual reality; wireless communications; performance measurements

1. Introduction

Virtual reality (VR) has become a technology that helps our daily lives [1]. From a simple example of catching a virtual monster in a real backyard setting, to an application that places virtual furniture in a room via a smartphone equipped with radar sensors, it is widely used in augmented reality (AR) [2]. This is based on the development of vision sensors and the lowering of prices for Lidar sensors, which are fitted to mobile devices. Currently, research is being conducted on experience-sharing AR in which users can experience VR in real worlds [3–5]. For example, one user's behavior with a virtual object means that they can observe it as it is when others look at the virtual object. Existing augmented and virtual reality is heavily influenced by device dependencies [6], that is, when a device already runs an application, it satisfies all the computing and communication requirements desired to run the application; therefore, no consideration is given to computers and communication networks [6]. However, the situation is different in AR services when the AR experience is shared. It requires a lot of bandwidth because AR shares the point cloud information for a virtual object by default [7–9]. Users must ensure a certain level of data rate requirements or higher for normal services. The increased number of users leads to a reduction in data rates due to the limited bandwidth over wireless links, which results in quality degradation. This paper identifies fluctuations in terms of data rates when point cloud information is shared in real-time authorized reality environments over IEEE 802.11ac wireless networks. In addition, this paper observes and analyzes frames-per-second (FPS) and Jitters, which are practical performance evaluation indices in real-time video streaming. The contribution of this paper can be summarized in three ways.

- This paper conducts the measurement study for real-world VR streaming services (i.e., streaming from servers to end-users) in terms of data rates, FPS, and Jitters with the Unity software library. Notice that the data rates, FPS, and Jitters are major real-world video streaming performance evaluation criteria that are widely and actively used [10,11].
- In addition, this paper also analyzes the FPS and Jitter fluctuations based on the data rate dynamics over IEEE 802.11ac wireless networks.
- Lastly, to the best of our knowledge, this work is the first to conduct a measurement study for defining the fundamental limits of VR streaming over IEEE 802.11ac wireless links. These kinds of measurement studies are definitely and essentially required for multimedia streaming research, e.g., [12]. Note that the introduction to measurement study trends are presented in Section 2.

The rest of this paper is organized as follows. Section 2 summarizes the related work of this measurement study. In addition, Section 3 presents the deep-dive measurement study results for VR content streaming in IEEE 802.11ac wireless networks. In addition, the results are discussed in many aspects. Lastly, Section 4 concludes this work and presents future research directions.

2. Related Work

The related research results for the measurements study of VR contents streaming over wireless networks can be classified into three main categories. First of all, in order to transfer and deliver point cloud information in real time, many research activities have been conducted for the compression of point cloud information because it involves an enormous amount of data [13,14].

The proposed algorithm in [13] introduces how to extend pointer clouds in decoders by encoding data structural differences. They eliminate spatial and temporal redundancies between continuous point clouds by differentially encoding the changes within the octet data structure of the point cloud. Furthermore, it presents Voxel-local point detail coding, which reduces the computational and memory requirements to enable online compression by increasing the point precision.

The proposed algorithm in [14] aims at the use of point clouds in autonomous driving to investigate accidents or apply them to various applications. They tried to solve the limitation problem due to the data size of the pointer cloud via a deep learning-based streaming point cloud compression method that can be performed in real time. It efficiently reduces temporal redundancy, using U-net with decoders and encoders and the preprocessing of data. Furthermore, when collecting data, 3D cloud information is stored without loss in a 2D matrix to reduce redundancy in space. During the process of collecting data, streaming occurs in real time, thereby reducing any temporal redundancy. Therefore, it is demonstrated through experiments that the method mentioned in [14] is more efficient than the previously proposed Octree compression, MPEG-based compression, and SLAM-based compression methods. The above studies have introduced ways to compress data in point clouds themselves in their own distinct way.

The proposed algorithm in [15] collects 3D spatial data using Microsoft Kinect sensors and utilizes the point cloud library (PCL) to process and filter 3D spatial data. It introduces a method for measuring and comparing FPS by adjusting the filtering time and filter ratio on PCL when streaming point clouds. The filters used in this paper are the Voxel Grid filter, Approximate Voxel Grid filter, and Pass-Through filter, which removes non-infinite points and all points outside the specified interval for the specified field. A Voxel Grid filter is a filter that creates a 3D Voxel Grid across the entire input cloud and downsamples points within each individual Voxel based on its center. The Approximate Voxel Grid filter is a filter that approximates the centroid using a hashing function in the Voxel Grid filter to speed up the process while sacrificing accuracy. Experiments showed that the average

FPS of Pass-Through filters was the highest, thereby introducing a new study method for screening filters with the highest FPS when point clouds are streamed using several kinds of filters.

Furthermore, in order to improve the robustness of multimedia content streaming over wireless networks, multi-path end-to-end contents delivery is also actively studied [16–18]. In addition, the literature verifies that the multi-path algorithms outperform their comparing algorithms [16–18].

Finally, an efficient method is proposed for streaming high quality 3D clouds [19]. In this study, the authors run a pilot study for dynamic adaptive point cloud streaming, and conduct a study that extends the notion of dynamic adaptive streaming over HTTP to DASH-PC, which is a bandwidth and view-aware point cloud streaming system. The bandwidth problem that arises with high quality point cloud is solved by utilizing DASH-PC. The experimental results show that adaptation could significantly save the point cloud streaming bandwidth, improve the rendering performance, and have a low impact on quality. The experimental results also show that through adaptation, the point cloud streaming bandwidth can be greatly reduced and the rendering performance can be improved, without impacting the quality of service. Therefore, a dynamic point cloud streaming adaptation technique to address the high bandwidth and processing requirements of transmission and rendering of dense point clouds is obtained from this work.

In addition, this paper considers the measurement study of real-time VR contents streaming over wireless networks. The measurement study is *de facto* one of the major research fields that defines the fundamental limits of network and video streaming performance [20–26]. The measurement study has been widely and actively conducted. In wire-line internet-based multimedia streaming, various types of measurement studies have been conducted in Youtube video streaming [20,21], Netflix/Hulu video streaming [22], real-time video games [23], and audio/text-sequence streaming, such as Skype, iChat, and Google+ [24]. In wireless networks, measurement studies have been also performed in LTE networks [25,26]. These measurement studies are all based on their specific interests; however, none of them do not consider VR-specific real-time video streaming features and characteristics.

The related work contributions discussed in this section are all considered link-level performance estimation. These link-level performance evaluation study results are for defining the fundamental limits of real-time wireless VR streaming. However, the measurement-based performance study results in this paper additionally consider the computation impacts of real-time VR contents encoding/decoding and compression. Therefore, it is obvious that the presented measurement-based performance study results in this paper are much more useful for real-world VR network designers and engineers.

3. Measurement Study of AR/VR Streaming over IEEE 802.11ac Wireless Networks

This paper focuses on the delay measurement study of VR streaming services in terms of data rates as the number of users increases. The data rates are measured to quantify the quality degradation as more users connect to the 3D streaming service. The data rates must be measured in order to obtain the minimal requirements for maximum user experience.

3.1. Measurement and Experimental Setup

To emulate a realistic environment, data rates of AR/VR streaming services are configured over a IEEE 802.11ac wireless network condition. The experimental setup, where it depicts the measurement of data, is constructed as shown in Figure 1. As shown in Figure 1, Azure Kinect monitors its surrounding as 3D scenes and passes the information to Unity in the form of point clouds.

At the system model level, the point cloud information obtained from Azure Kinect is transferred to Unity. Azure Kinect is a PC peripheral device with an artificially intelligent sensor produced by Microsoft for computer vision. Unity is a widely used program to create 3D and 2D interactive content for animation, architectural visualization and VR.

With the point cloud information received from the camera, we construct a 3D scene of its real-time recorded environment. Unity also delivers high quality 3D scene information to servers created in Webrtc via render streaming and high definition render pipeline (HDRP). Webrtc allows 3D scenes from Unity to be streamed in real time to the client. The client sends control information (pitch, roll, yaw) to Unity through a data channel, which is once again produced by Webrtc.

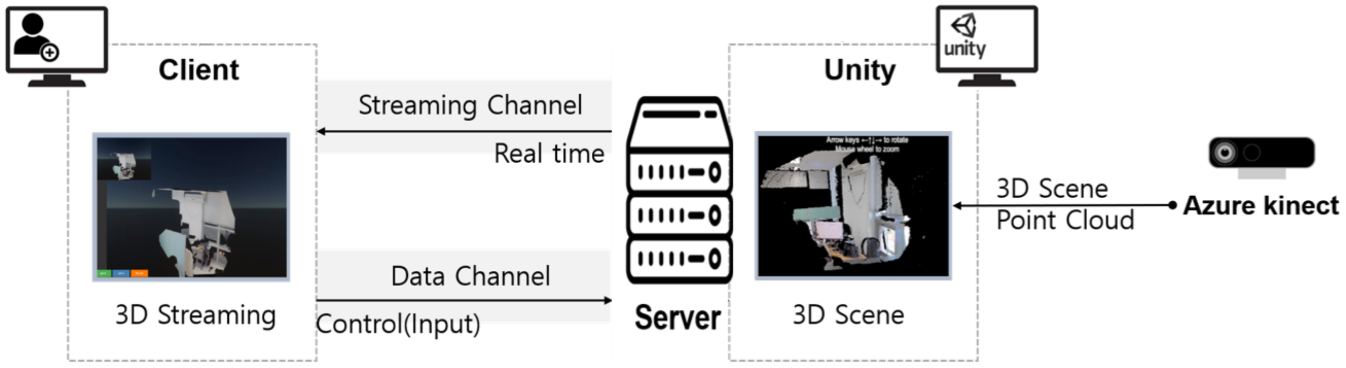

Figure 1. System model.

This section presents the experiment setup (refer to Section 3.1), implementation of data rate, Jitter, and FPS (refer to Section 3.2, Section 3.3, and Section 3.4, respectively) and the analysis of the proposed correlation (refer to Section 3.5). The hardware and software setup is also constructed in this section.

In terms of hardware, we employed an NVIDIA 2080Ti station equipped with 4 × Tesla V100 GPUs (a total of 128 GB memory available) and an Intel Xeon E5-2698 v4 2.2 GHz CPU with 20 cores (256 GB system memory available in total). Note that the details of hardware information for connected users is presented in Table 1. In terms of software, we utilized Azure Kinect viewer v1.4.1 to obtain 3D views of the captured scene and Unity version 2021.1.5f1 and Webrtc version 2.2.1 for real-time streaming.

Table 1. Simulation setup.

Hardware/Software	Specification
CPU	Intel(R) Core(TM) i5-7500 CPU @ 3.40 GHz Intel(R) Core(TM) i7-9700K CPU @ 3.60 GHz AMD Ryzen 7 1800X 8-core processor AMD® Ryzen threadripper 3970× 32-core processor × 64 Intel(R) Core(TM) i7-10700K CPU @ 3.80 GHz
GPU	NVIDIA GeForce GTX 1060 3 GB NVIDIA GeForce GTX 1660 NVIDIA GeForce GTX 1050 NVIDIA Corporation TU104 GeForce RTX 2080 SUPER Intel(R) UHD Graphics 630
RAM	16 GB, 64 GB, 128 GB
Unity	Unity version 2021.1.5f1
Azure kinect	Azure kinect viewer version 1.4.1
Webrtc	Webrtc version 2.2.1
Matlab	Matlab version 2021a
Operating system	Windows 10
Render streaming	Unity Render Streaming version version 2.0.2

First, the Azure Kinect depth camera is installed inside the Institute of Artificial Intelligence Engineering building at Korea University to capture the internal structure of the room as shown in Figure 2. The camera produces a 2D depth map of the footage taken. The same 2D depth map of the camera's coordinate system is converted into a 3D point cloud to present a 3D representation of the captured scene. Within the camera, 3D scenes are converted into point cloud, which is then compressed and processed using PCL. In order to stream the represented 3D scenes in real-time, the camera is linked with Unity and directly inputs the point cloud information.

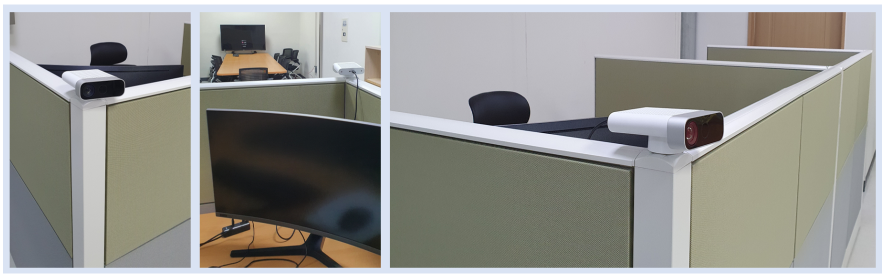
Figure 2. Experimental settings.

The implementation conducted using Unity is depicted in Figure 3, where communication indicators, such as the server's FPS and bit-rates, can be determined. In addition, the render streaming code written by Unity Script established a streaming environment, including resolution, camera adjustment angle, and number of point clouds. The 3D scene, converted from the 2D depth map using the point cloud, is shown in Figure 4. At the same time, the render streaming asset, with version v2.0.2, is utilized within Unity. Two render streaming cameras are implemented to connect with multiple users and to apply the user's request, which involve screen rotation due to touch and mouse manipulation, to other users. During streaming, the resolution of 1280*720 is maintained, using HDRP in Unity, to allow a consistent streaming experience with high resolution.

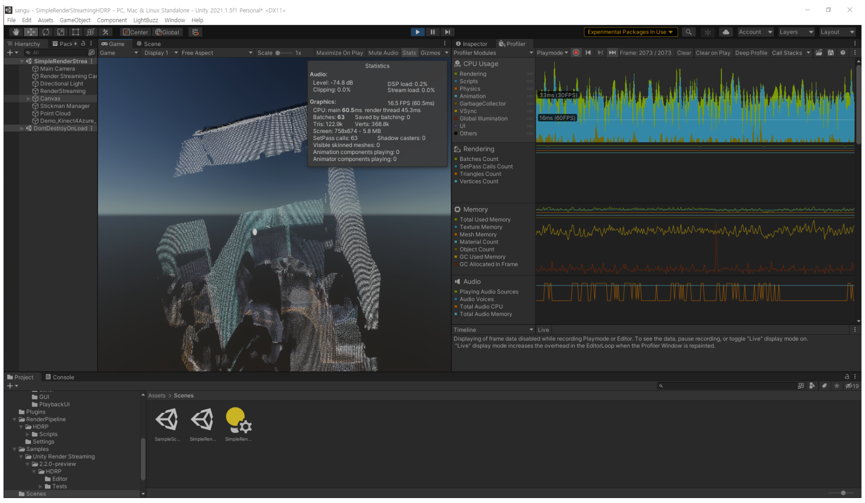
```
Statistics
Audio:
  Level: -74.8 dB                DSP load: 0.2%
  Clipping: 0.0%                 Stream load: 0.0%
Graphics:                        16.5 FPS (60.5ms)
  CPU: main 60.5ms  render thread 45.3ms
  Batches: 63      Saved by batching: 0
  Tris: 122.9k     Verts: 368.8k
  Screen: 758x674 - 5.8 MB
  SetPass calls: 63          Shadow casters: 0
  Visible skinned meshes: 0
  Animation components playing: 0
  Animator components playing: 0
```

Figure 3. Simulation model.

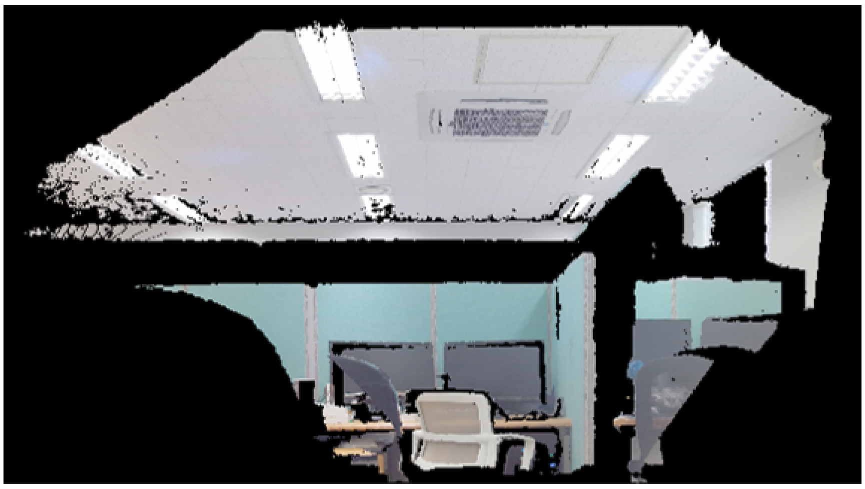

Figure 4. Unity 3D scene.

Our proposed model differs significantly from the existing point cloud streaming methods, where Figure 3 is implemented in Unity. By comparison, Figure 4 is streamed to a much higher resolution point cloud that is enabled through the implementation of HDRP. Unity runs a web server called Webrtc to connect with users in real time. Connecting and streaming to the user is represented in Figure 5. When the users and server are connected, Webrtc measures the bit-rate, Jitter buffer performance, number of packets received and dropped, and video frame rate.

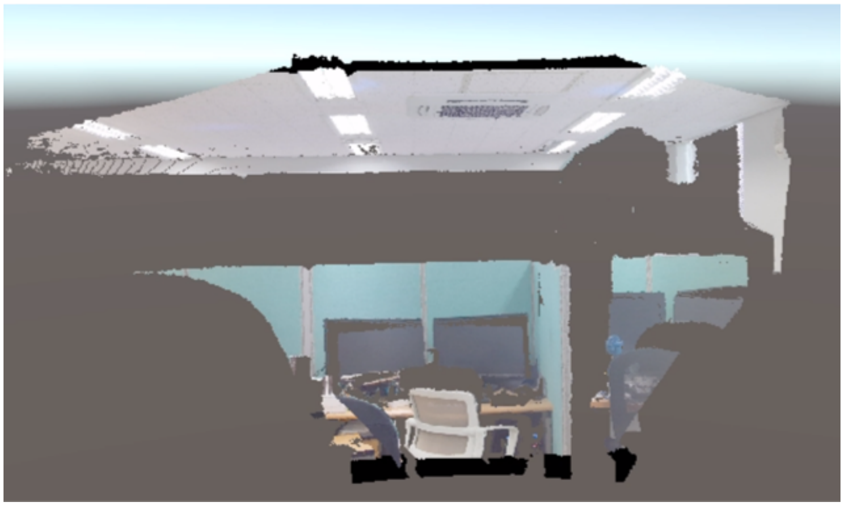

Figure 5. Streaming 3D scene.

There are three main types of experiments conducted in this paper. The first experiment is devised to understand the changes in the data rate to compare the quality of communication as the number of users increases. The second experiment is conducted to monitor the changes in the performance indicators, such as Jitter, FPS, or Frame Dropped, depending on the number of users. The final experiment measures the variations in the data rate conditional to the changes in those performance indicators.

The indicators used to compare the performance of communication depending on the user number are Jitter and FPS. Jitter portrays the change in latency and delay in communication situations. It is an adequate performance evaluation factor, as it determines how smoothly communication proceeds, depending on the stabilization state of Jitter [27]. FPS is the most common performance indicator used to determine the state of the communication. If the real-time communication situation is not stable, the number of frames on the screen being transmitted is reduced and the FPS is lowered. Therefore, we select FPS as the indicator to evaluate the communication performance. To support the assessment using the FPS indicator, we consider an auxiliary indicator called Frame Dropped.

The experimental results are obtained by extracting graphs as shown in Figures 6–9, where all experiments are repeatedly conducted with varying the number of users. Figures 6–9 display the data information required to analyze the communication status using the above indicators depending on the number of users. The distance between the connected PC and other users is 2.5 m. It should be noted that the connection of users to the PC occurs only when the bit-rates per second (bps) has stabilized to some extent, which is represented by the red dotted circles in Figure 6. In Figure 6, the orange dotted circles depict the times the PC connects to users. Figure 6 is the value of the data rate according to the number of users connected; Figure 7 represents the change in Jitter according to the number of users connected to the server. Lastly, Figures 8 and 9 represent the change in FPS and Frame Dropped, respectively, depending on the number of users connected.

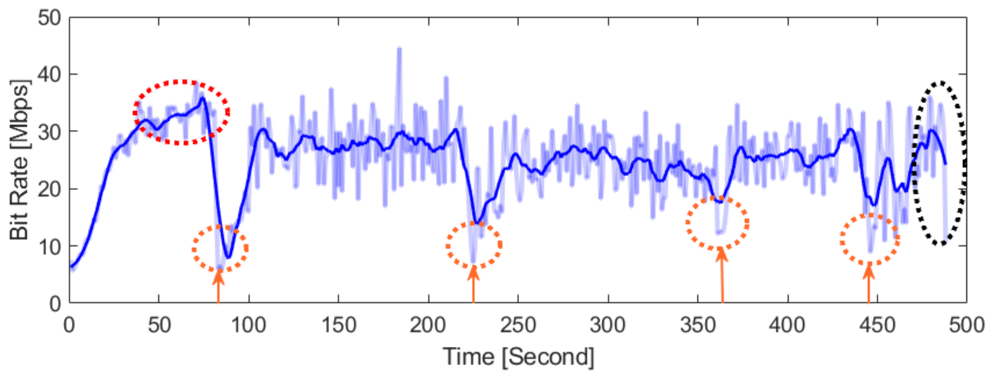

Figure 6. Bit Rate measurement graph by number of users connected.

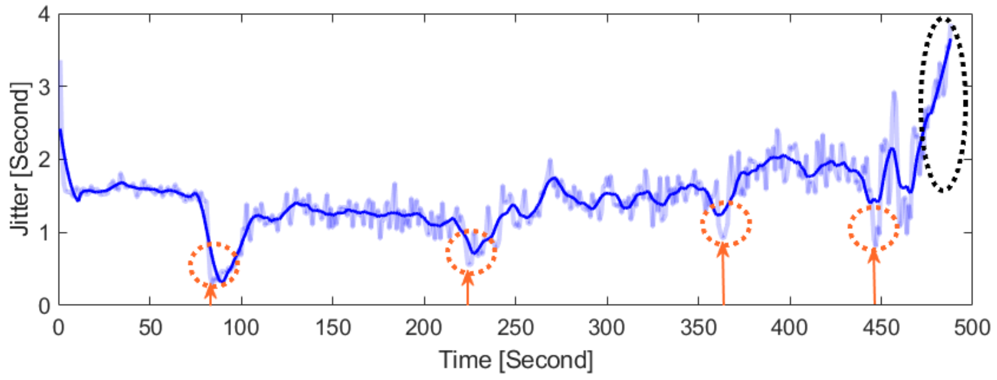

Figure 7. Jitter measurement graph by number of users connected.

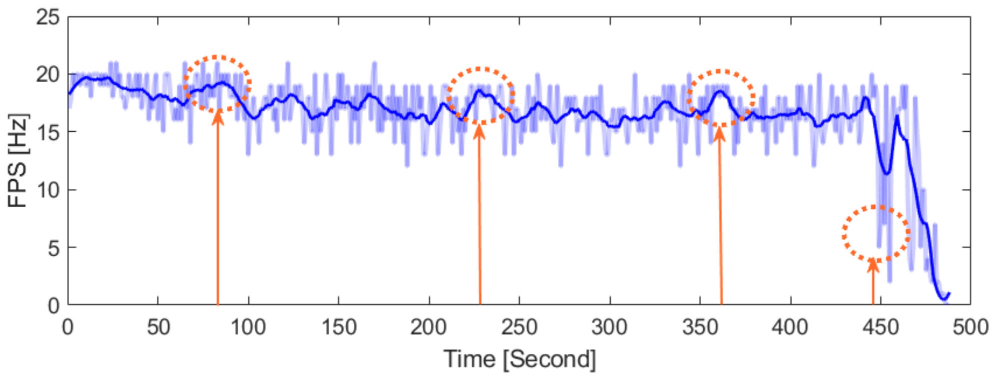

Figure 8. FPS measurement graph by number of users connected.

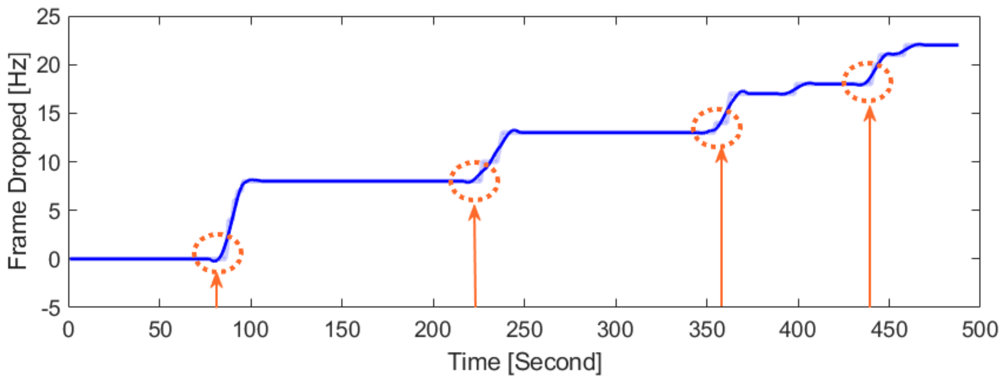

Figure 9. Frame Dropped measurement graph by the number of connected users.

3.2. Measurement Results–Data Rate

As the number of users increases, the change in data rate is summarized in Table 2. The value of the data rate shown in the table is the average of the values obtained from three experiments. This averaged data rate value is acquired when the bps is stable (i.e., the positions indicated with the red dotted circle) in Figure 6. The orange arrows in Figure 6 denote the time when a new connection between a user and PC is made. During those positions, a sharp decline in bit rate, which is essentially the same concept as the data rate, can be observed. As more users connect to a server, the fixed amount of the data rate needs to be distributed among those users, thereby decreasing the assigned data rate to each user.

Table 2. Data rate by the number of users connected (unit: Mbps).

Number of Users	User 1	User 2	User 3	User 4	User 5	Average
1	38.742	-	-	-	-	38.742
2	29.626	21.144	-	-	-	25.385
3	28.318	21.117	19.085	-	-	22.840
4	26.807	18.698	18.547	16.944	-	20.249
5	25.739	18.597	17.458	15.637	15.4729	18.581

3.3. Measurement Results—Jitter

Similarly, the change in the Jitter value as the number of users increases is shown in Table 3. The level of the Jitter declines as more users connect to the server, as shown in Figure 7. The fall in Jitter levels indicates that the bitrate rises, resulting in a reduction in latency between users and servers. When the user is in the process of connecting to the server, the Jitter value becomes unstable as depicted by the orange dotted circles in Figure 7. The black dotted circles in Figure 7 represent the moment when the user disconnects from the server. As seen in this part of the graph, the Jitter value soars, and the bitrate levels drop to a minimum.

Table 3. Jitter by number of users connected (unit: second).

Number of Users	User 1	User 2	User 3	User 4	User 5	Average
1	1.550	-	-	-	-	1.550
2	1.611	2.471	-	-	-	2.041
3	1.973	2.524	2.928	-	-	2.475
4	3.676	3.847	3.943	4.534	-	3.999
5	4.118	4.446	4.715	5.1482	6.198	4.925

The Jitter change with respect to the data rate is illustrated in Figure 10. This graph, with data rate values from 0 to 40, shows that the Jitter and data rate are inversely proportional. The red crosses in Figure 10 are the actual data points measured. These values are regressed to predict the line of best fit. The equation for the regression curve, shown in black, is shown as an equation of curves in the form of $y = 5.958 \times e^{-0.0229x}$ exponential. As can be seen by the regression equation, Jitter and the data rate have a inverse relationship.

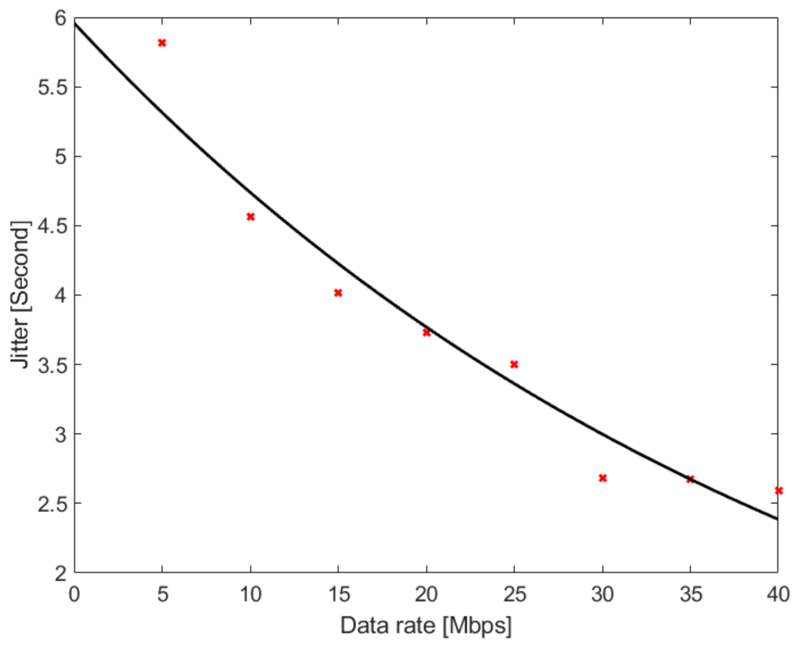

Figure 10. Jitter according to data rate.

3.4. Measurement Results–FPS

Comparably, the change in the performance indicator FPS is also explored in Table 4 as the number of users increases. The values in the table indicate that the higher the number of users connected to the server, the lower the quality of communication provided. In addition, the Frame Dropped indicator increases with the number of connected users as shown in Figure 9. Furthermore, the change in FPS with data rates is depicted in Figure 11. This graph indicates that the data rate and FPS increase proportionately. The red cross points displayed in Figure 11 is a measurement of the actual data, and these values were regressed to predict the line of best fit, represented in the form of the black test curve. The equation for the regression curve is shown as an equation of curves in the form of $y = 5.499 \times e^{0.0255x}$ exponential. In other words, the FPS and data rate have a proportional relationship.

Table 4. FPS by the number of users connected users (unit: Hz).

Number of Users	User 1	User 2	User 3	User 4	User 5	Average
1	17.333	-	-	-	-	17.333
2	15.333	15.000	-	-	-	15.166
3	14.667	14.333	14.000	-	-	14.333
4	12.667	12.333	11.667	11.000	-	11.917
5	12.000	11.667	11.000	10.667	8.000	10.667

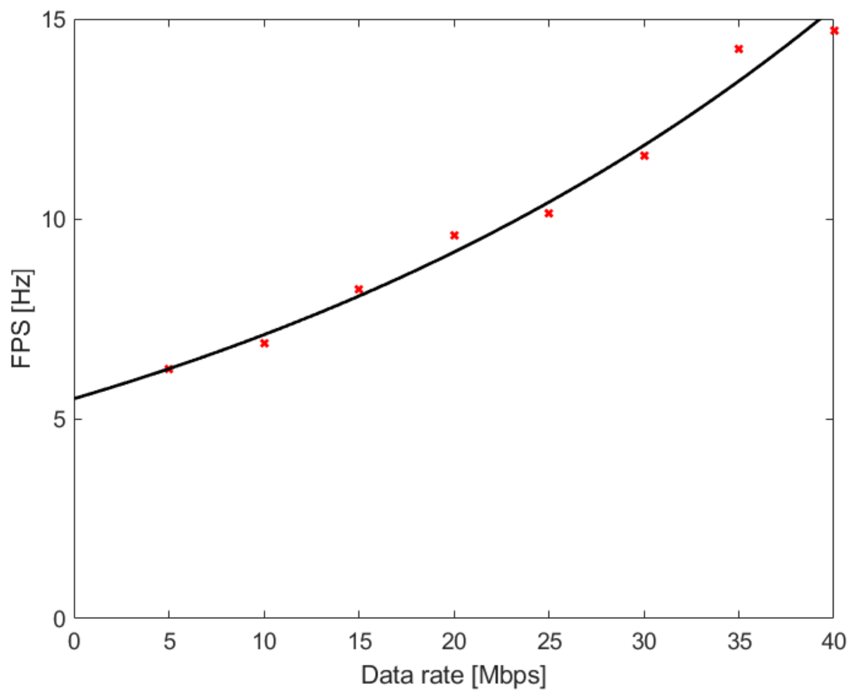

Figure 11. FPS according to data rate.

3.5. Measurement Results—Performance Indicator Analysis

In this section, we intensively measure and analyze the real-time VR streaming performance over IEEE 802.11ac wireless links in terms of data rate, Jitter, and FPS in various communication situations. The correlation of each indicator according to the data rate is shown in the forms of $y = 5.958 \times e^{-0.0229x}$ and $y = 5.499 \times e^{0.0255x}$, respectively, as analyzed in Sections 3.3 and 3.4, respectively. The correlation between Jitter and FPS is clearly illustrated in Figure 12, based on our measurement study. The red crosses representation of Figure 12 is the measurement of the actual data and the regression of these values are calculated. As the FPS value does not fall below 6, the data are predicted to be in a straight black line. Regression results show the linear equation of $y = -0.323x + 6.989$. The correlation coefficient is -0.835, which shows a strong negative correlation between Jitter and FPS. This shows that each indicator is correlated and there is the minimum amount of the data rate that can maintain optimal communication depending on the number of users connected.

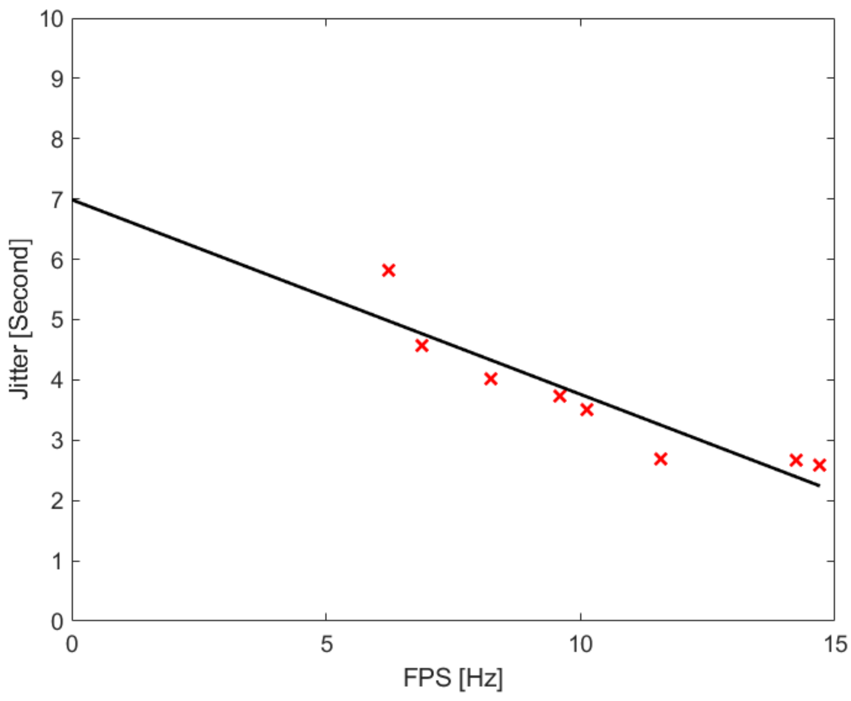

Figure 12. Jitter according to FPS.

Lastly, Tables 2–4 show that the performance in terms of the data rate, Jitter, and FPS, depends on the number of users. For the case of standalone single user, it can utilize maximum communication resources; therefore, it can maintain good performance in terms of the given performance metric. If we have multiple users, the communication resources should be shared, i.e., the performance will be degraded. In terms of Unity software's render streaming setting, the communication resource allocation priority will be ordered from user 1 to user 5. Thus, we can observe the measurement results as shown in the Tables 2–4.

4. Conclusions and Discussion

As the pandemic creates more restrictions and lock-downs in neighborhoods, more and more people engage in 3D video streaming to interact with each other at a more intimate level. This paper searched for an effective way to maintain the quality of streaming of 3D scenes captured by a camera as more users access the service. Therefore, this paper conducted several experiments to determine our target data rate with the best communication network status for a pleasant 3D streaming experience among multiple clients. The experiments show that as the number of connected users increases, the data rate received by each user decreases. This means that there is a limited amount of data rate at the server that can be assigned to users; when the number of users increases, the data rate provided to existing and new users decreases, resulting in inevitable quality degradation.

Hence, in a setting where the number of users increases in the same communication state, we measured FPS, Jitter, and cumulative Frame Dropped values, which all represent the quality of communication. Consequently, an increase in the number of connected users results in a decrease in FPS, an increase in Jitter, and an increase in the cumulative number of Frame Dropped. This shows that the more users connect to the server, the lower the quality of communication. To determine the optimal communication environment based on data rate allocated for total users, we need to know the relationship between performance indicators. Therefore, we first measured the change in Jitter and FPS with data rate. As the data rate increased, Jitter declined, as shown in Figure 10. On the other hand, as the data rate increased, FPS increased along the curve of Figure 11. The relationship between Jitter and FPS has a strong negative correlation with a correlation coefficient of −0.835, like that shown in Figure 12. Thus, we conclude with the fact that there is a correlation between performance indicators, which we can leverage to find the minimal data rate that maintains the communication quality, even as the number of connected users increases.

This study is based on the field of Extended Reality (XR). The above study measures the state of communication that changes as the number of users increases when point clouds are shared in real-time authorized reality environments. The data rate and the performance metrics that immediately reflect the quality of user experience, Jitter and FPS, are measured and analyzed. As our future research directions, the following topics are worthy of consideration.

- This work can be developed into a study of the optimization of the minimum data rate and communication indicators to maintain the quality of real-time communication in an XR environment. It can also be extended to evolve into the commercialization of real-time reality streaming systems.
- In order to improve the communication efficiency over the fundamental limit measurement study of wireless VR streaming as shown in this paper, the algorithms for dealing with spatial and temporal redundancies under the consideration of sparsity and disorder are desired. It is required to jointly consider the aspects on top of our wireless link measurement study in this paper.
- Moreover, we conducted the measurement study in an office environment for this paper. For the extension, we will conduct this type of measurement study in various environments, e.g., campus environment, outdoor environment, and smart factory environment for digital twin applications.
- Lastly, we can consider various wireless communication technologies, such as 60 GHz millimeter-wave networks [28–30].

Author Contributions: W.J.Y., G.L., S.J., J.K. (Jiyeon Kim) and S.H. were the main researchers who initiated and organized the research reported in the paper, and all authors including Y.J.H., Y.K.L. and J.K. (Joongheon Kim) were responsible for writing the paper and analyzing the simulation results. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Samsung Electronics Co., Ltd. (IO201208-07855-01).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. de Souza Cardoso, L.F.; Mariano, F.C.M.Q.; Zorzal, E.R. A survey of industrial augmented reality. *Comput. Ind. Eng.* **2020**, *139*, 106159. [CrossRef]
2. Na, W.; Dao, N.N.; Kim, J.; Ryu, E.S.; Cho, S. Simulation and measurement: Feasibility study of Tactile Internet applications for mmWave virtual reality. *ETRI J.* **2020**, *42*, 163–174. [CrossRef]
3. Unal, M.; Bostanci, E.; Sertalp, E. Distant augmented reality: Bringing a new dimension to user experience using drones. *Digit. Appl. Archaeol. Cult. Herit.* **2020**, *17*, e00140. [CrossRef]
4. Ghazwani, Y.; Smith, S. Interaction in Augmented Reality: Challenges to Enhance User Experience. In Proceedings of the 2020 4th International Conference on Virtual and Augmented Reality Simulations, Sydney, Australia, 14–16 February 2020; pp. 39–44.
5. Yi, J.; Kim, S.; Kim, J.; Choi, S. Supremo: Cloud-Assisted Low-Latency Super-Resolution in Mobile Devices. *IEEE Trans. Mob. Comput.* **2021**. [CrossRef]
6. Sereno, M.; Wang, X.; Besançon, L.; McGuffin, M.J.; Isenberg, T. Collaborative Work in Augmented Reality: A Survey. *IEEE Trans. Vis. Comput. Graph.* **2020**. [CrossRef] [PubMed]
7. Capraro, F.; Milani, S. Rendering-aware point cloud coding for mixed reality devices. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 3706–3710.
8. Gao, L.; Bai, H.; Lee, G.; Billinghurst, M. An oriented point-cloud view for MR remote collaboration. In Proceedings of the SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications, Macao, 5–8 December 2016; pp. 1–4.
9. da Silva Cruz, L.A.; Dumić, E.; Alexiou, E.; Prazeres, J.; Duarte, R.; Pereira, M.; Pinheiro, A.; Ebrahimi, T. Point cloud quality evaluation: Towards a definition for test conditions. In Proceedings of the 2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX), Berlin, Germany, 5–7 June 2019; pp. 1–6.
10. Kim, J.; Caire, G.; Molisch, A.F. Quality-Aware Streaming and Scheduling for Device-to-Device Video Delivery. *IEEE/ACM Trans. Netw.* **2016**, *24*, 2319–2331. [CrossRef]
11. Choi, M.; Kim, J.; Moon, J. Wireless Video Caching and Dynamic Streaming Under Differentiated Quality Requirements. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 1245–1257. [CrossRef]

12. Kim, J.; Lee, W. Feasibility Study of 60 GHz Millimeter-Wave Technologies for Hyperconnected Fog Computing Applications. *IEEE Internet Things J.* **2017**, *4*, 1165–1173. [CrossRef]
13. Kammerl, J.; Blodow, N.; Rusu, R.B.; Gedikli, S.; Beetz, M.; Steinbach, E. Real-time compression of point cloud streams. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 14–18 May 2012; pp. 778–785.
14. Tu, C.; Takeuchi, E.; Carballo, A.; Takeda, K. Real-Time Streaming Point Cloud Compression for 3D LiDAR Sensor Using U-Net. *IEEE Access* **2019**, *7*, 113616–113625. [CrossRef]
15. Moreno, C.; Li, M. A comparative study of filtering methods for point clouds in real-time video streaming. In Proceedings of the World Congress on Engineering and Computer Science, San Francisco, CA, USA, 19–21 October 2016; Volume 1, pp. 388–393.
16. Wu, J.; Cheng, B.; Yuen, C.; Shang, Y.; Chen, J. Distortion-Aware Concurrent Multipath Transfer for Mobile Video Streaming in Heterogeneous Wireless Networks. *IEEE Trans. Mob. Comput.* **2015**, *14*, 688–701. [CrossRef]
17. Wu, J.; Yuen, C.; Cheng, B.; Wang, M.; Chen, J. Streaming High-Quality Mobile Video with Multipath TCP in Heterogeneous Wireless Networks. *IEEE Trans. Mob. Comput.* **2016**, *15*, 2345–2361. [CrossRef]
18. Wu, J.; Yuen, C.; Wang, M.; Chen, J. Content-Aware Concurrent Multipath Transfer for High-Definition Video Streaming over Heterogeneous Wireless Networks. *IEEE Trans. Parallel Distrib. Syst.* **2016**, *27*, 710–723. [CrossRef]
19. Hosseini, M.; Timmerer, C. Dynamic adaptive point cloud streaming. In Proceedings of the 23rd Packet Video Workshop, Amsterdam, The Netherlands, 12–15 June 2018; pp. 25–30.
20. Cheng, X.; Liu, J.; Dale, C. Understanding the Characteristics of Internet Short Video Sharing: A YouTube-Based Measurement Study. *IEEE Trans. Multimed.* **2013**, *15*, 1184–1194. [CrossRef]
21. Zou, S.; Wang, Q.; Ge, J.; Tian, Y. Peer-Assisted Video Streaming With RTMFP Flash Player: A Measurement Study on PPTV. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *28*, 158–170. [CrossRef]
22. Adhikari, V.K.; Guo, Y.; Hao, F.; Hilt, V.; Zhang, Z.L.; Varvello, M.; Steiner, M. Measurement Study of Netflix, Hulu, and a Tale of Three CDNs. *IEEE/ACM Trans. Netw.* **2015**, *23*, 1984–1997. [CrossRef]
23. Xue, Z.; Wu, D.; He, J.; Hei, X.; Liu, Y. Playing High-End Video Games in the Cloud: A Measurement Study. *IEEE Trans. Circuits Syst. Video Technol.* **2015**, *25*, 2013–2025. [CrossRef]
24. Xu, Y.; Yu, C.; Li, J.; Liu, Y. Video Telephony for End-Consumers: Measurement Study of Google+, iChat, and Skype. *IEEE/ACM Trans. Netw.* **2014**, *22*, 826–839. [CrossRef]
25. van der Hooft, J.; Petrangeli, S.; Wauters, T.; Huysegems, R.; Alface, P.R.; Bostoen, T.; De Turck, F. HTTP/2-Based Adaptive Streaming of HEVC Video Over 4G/LTE Networks. *IEEE Commun. Lett.* **2016**, *20*, 2177–2180. [CrossRef]
26. Xu, Y.; Xiao, Z.; Feng, H.; Yang, T.; Hu, B.; Zhou, Y. Modeling Buffer Starvations of Video Streaming in Cellular Networks with Large-Scale Measurement of User Behavior. *IEEE Trans. Mob. Comput.* **2017**, *16*, 2228–2245. [CrossRef]
27. Hakak, S.; Latif, S.A.; Anwar, F.; Alam, M.K. Impact of key factors on average jitter in MANET. In Proceedings of the International Conference on Systems Informatics, Modelling and Simulation (SIMS), Sheffield, UK, 29 April–1 May 2014; pp. 179–183.
28. Jung, S.; Kim, J.; Levorato, M.; Cordeiro, C.; Kim, J.H. Infrastructure-Assisted On-Driving Experience Sharing for Millimeter-Wave Connected Vehicles. *IEEE Trans. Veh. Technol.* **2021**. [CrossRef]
29. Choi, M.; Kim, J.; Moon, J. Dynamic Power Allocation and User Scheduling for Power-Efficient and Delay-Constrained Multiple Access Networks. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 4846–4858. [CrossRef]
30. Kim, J.; Molisch, A.F. Fast Millimeter-Wave Beam Training with Receive Beamforming. *J. Commun. Netw.* **2014**, *16*, 512–522. [CrossRef]