Interacting With New York City Data by HoloLens Through Remote Rendering

Zijian Long
University of Ottawa

Haiwei Dong
Huawei Technologies Canada

Abdulmotaleb El Saddik
University of Ottawa
Mohamed Bin Zayed University of Artificial Intelligence

***Abstract*—In the digital era, extended reality (XR) is considered the next frontier. However, XR systems are computationally intensive, and they must be implemented within strict latency constraints. Thus, XR devices with finite computing resources are limited in terms of quality of experience (QoE) they can offer, particularly in the cases of big 3D data. This problem can be effectively addressed by offloading the highly intensive rendering tasks to a remote server. Therefore, we proposed a remote rendering enabled XR system that presents the 3D city model of New York City on the Microsoft HoloLens. Experimental results indicate that remote rendering outperforms local rendering for the New York City model with significant improvement in average QoE by at least 21%. In addition, we clarified the network traffic pattern in the proposed XR system developed under the OpenXR standard.**

HOLOLENS AND REMOTE RENDERING

■ **Extended reality (XR)** is a term referring to all real-and-virtual combined environments and human–machine interactions.[1] It subsumes the entire spectrum of realities assisted by immersive technology, such as virtual reality (VR), augmented reality (AR), and mixed reality (MR).[2] A VR system blocks out the outside world and simulates a realistic environment where participants can move around and view, grab, and reshape objects in a virtual environment.[3] As opposed to VR, AR provides a physical real-world

Digital Object Identifier 10.1109/MCE.2022.3165961
Date of publication 8 April 2022; date of current version 17 August 2022.

2162-2248 © 2022 IEEE Published by the IEEE Consumer Technology Society

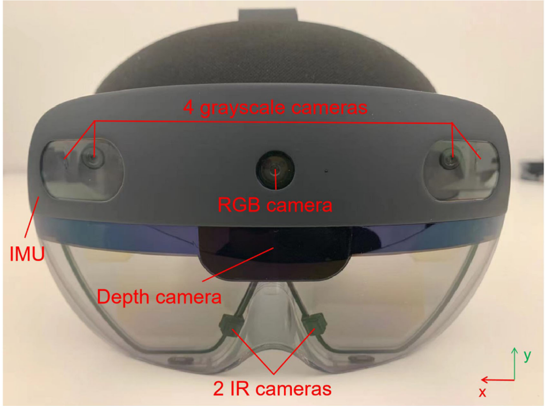


Figure 1. HoloLens 2 is equipped with multiple sensors to support hand tracking, eye tracking, and voice command for human understanding. It is also capable of understanding environment by six degrees of freedom (6 DoF) tracking and spatial mapping.

environment that has been enhanced with 3D virtual objects,[4] and MR is a hybrid reality where physical and digital objects coexist and interact in real time.[5] These XR technologies are redefining how people experience the world by blurring the line between the real and virtual. It is expected that the global XR market will reach 300 billion U.S. dollars by 2024.[6]

Recently, we have seen an increase in the development of XR technologies, with an increasing number of companies producing XR devices. A cutting-edge example of smart MR glasses is the Microsoft HoloLens 2. Compared with other MR devices, such as the HoloLens 1,[7] the HoloLens 2 provides a more immersive experience with a larger Field of View (52°) and a higher resolution (2048×1080 per eye). As shown in Figure 1, the HoloLens 2 employs eight cameras to facilitate human and environment understanding: four visible-light tracking cameras for real-time visual-inertial simultaneous localization and mapping, a depth camera for articulated hand tracking as well as spatial awareness, two infrared (IR) cameras for eye tracking, and an RGB color camera to create MR photos and videos for users. There are also inertial measurement units including an accelerometer to determine the linear acceleration along the *x*-, *y*-, and *z*-axis (right-hand rule), a gyroscope to determine rotations, and a magnetometer for absolute orientation estimation. These sensors lead to a large amount of data for the HoloLens 2 to transmit/process (see Table 1). For example, for the four grayscale cameras used for head tracking, the raw data rate is 294.9 Mbits/s with a frame rate of 30 Hz and a resolution of 640 × 480 using 8 bits to encode the RGB data of each pixel. Therefore, computing power is in high demand to support all of these tracking functions.

Although the HoloLens 2 is equipped with a holographic processing unit (HPU) to handle all computing tasks, the HPU may not be able to provide detailed models of specific scenarios where every detail matters, such as truck engines and city models. It is possible to solve these problems by remote rendering, which offloads complex rendering tasks to a rendering server with powerful computing capability. Users may experience a superior quality of experience (QoE)[8] when using remotely rendered systems compared to those rendered by the HoloLens 2 itself. Figure 2 demonstrates the proposed XR system architecture in which a user interacts with the 3D digital data. The HoloLens 2 detects the user's movement and sends it to the remote server through an uplink (UL) stream of the XR network. The server performs logic processing and renders corresponding video frames based on the user's movement. The encoded video frames are then transmitted back to the user through a downlink (DL) stream of the XR network.

CASE STUDY: AN XR SYSTEM TO INTERACT WITH HOLOGRAPHIC DATA OF NEW YORK CITY

System Setup and Data Acquisition

Our rendering server is equipped with the Windows 10 operating system, an Intel Core i9-11900F processor, 64 GB of random-access memory, and an NVIDIA GeForce RTX 3090 graphics card. For the HMD, the HoloLens 2 features the Qualcomm Snapdragon 850 Compute Platform, which comprises four gigabytes of LPDDR4x system memory. As a proof-of-concept, the holographic remoting player is installed on the HoloLens 2 to connect the HoloLens 2 to the server through a USB-C cable. As an example made by Unity, we developed our system using multiple auxiliary packages. An important package is OpenXR, which is a royalty-free, open

Table 1. Technical specifications of cameras that the Microsoft HoloLens 2 is equipped with.

	Cameras	FPS	Resolution	Format	Raw data
Head tracking	4 Grayscale	30	640 × 480	8-bit	294.9 Mbits
Hand tracking	1 Depth	45	512 × 512	16-bit	188.7 Mbits
Eye tracking	2 IR	–	–	–	–
Spatial awareness	1 Depth	1–5	320 × 288	16-bit	1.5–7.3 Mbits

(FPS stands for frames per second.) These cameras introduce a large number of data for the hololens 2 to process to support these functions.

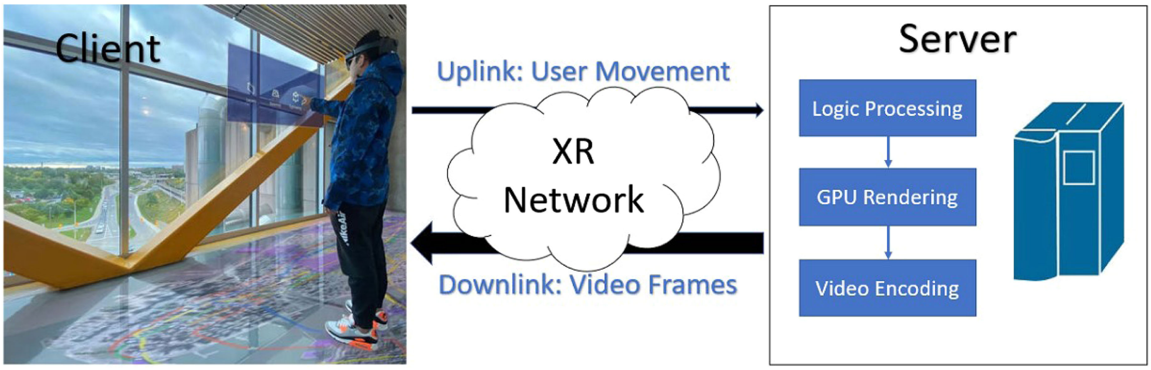


Figure 2. Architecture of our proposed system consisting of a user with the HoloLens 2, an XR network, and a rendering server. On the left, a user is wearing the HoloLens 2 to interact with the New York City data.

standard that provides high-performance access to XR platforms and devices. Nowadays, OpenXR has been supported by most of the industry-leading companies, such as Microsoft, Huawei, and Meta.

The New York City data was downloaded from ArcGIS Living Atlas of the World, the world's premier collection of geographic information.[9] The site provides live feeds and other content that assists us in understanding current events and infrastructure in the form of information layers. These information layers contain geographic information, such as maps and the distribution of buildings that can be imported directly into the system.

Interaction Design

We designed an interaction system where multiple interaction models are provided to allow users to naturally interact with holographic city data. Though there could be various effective and engaging interaction ways supported by the HoloLens 2,[10] we proposed three basic interaction models based on hand tracking, eye tracking, and voice commands in our system. With these natural interaction models, users can quickly learn how to manipulate objects within an XR environment without any difficulty.

A key feature of the HoloLens 2 is hand tracking, which allows the device to identify the user's hands and fingers and track them in the air. With fully articulated hand tracking, users can touch, grasp, and move holographic objects naturally. In our system, hand tracking supports two types of interactions: the near interaction and the far interaction. The near interaction is a method of controlling objects that are physically close to the user, allowing their hands to directly manipulate them. As an example, buttons are activated by simply pressing them. In addition, we provide a way to manipulate objects out of reach using the "point and commit" interaction, which is a unique interaction method in the XR world. It involves the ray shooting out of the user's palm and includes two stages: the pointing stage and the committing stage. In the former stage, users should reach out their hands with a dashed ray [see Figure 4(a)] while the end of the

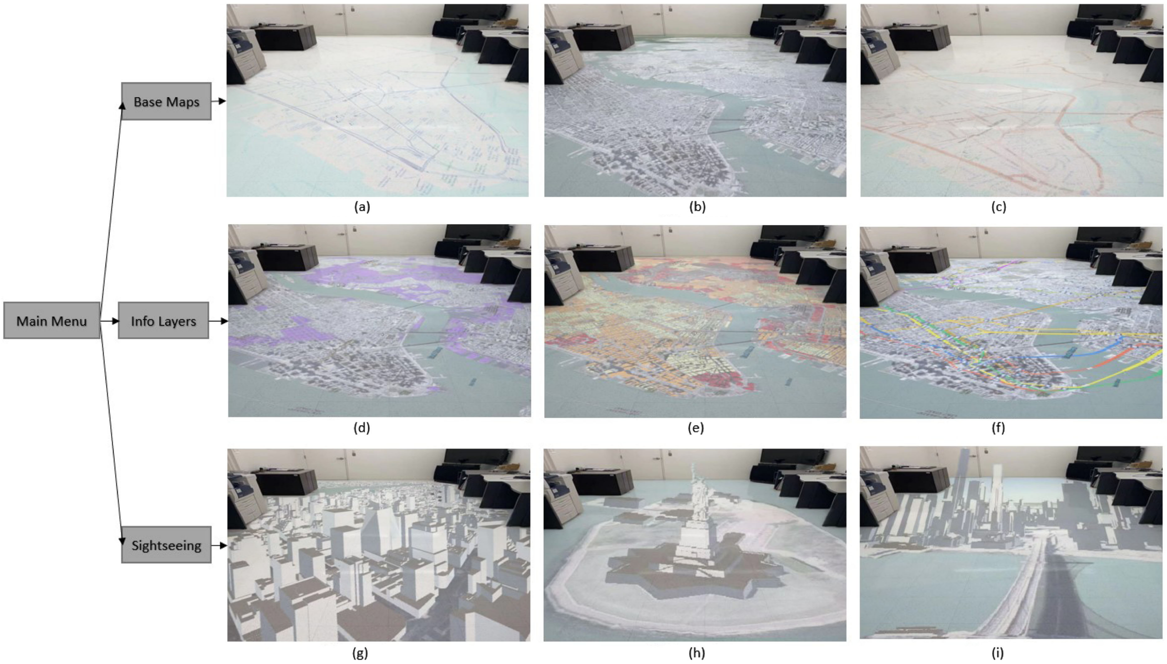


Figure 3. There are three sections in the main menu: the "Base Maps" section, "Info Layers" section, and "Sightseeing" section. We show three examples for each section: topography, imagery, streets for "Base Maps," industrial area, population density, subways for "Info Layers," and Times Square, Statue of Liberty, Brooklyn Bridge for "Sightseeing." (a) Topography. (b) Imagery. (c) Streets. (d). Industrial area. (e) Population density. (f) Subways. (g) Times Square. (h) Statue of Liberty. (i) Brooklyn bridge.

ray indicates the target object. In the latter stage, which can be triggered by the thumb and index finger [see Figure 4(b)], the ray becomes a solid line, allowing users to interact with 3D objects from a distance.

The HoloLens 2 is able to detect a user's gaze by using sensors observing at the eyes. This eye gaze indicates a signal for the user's focus and intent. Moreover, it is important for users to receive appropriate feedback and indication when interacting with the city data in our system. As an example, when a user gazes at a famous building in New York City in our system, the name of the building will appear as feedback. The other hands-free modality in our system to interact with holograms is using voice commands. Voice commands are usually used in conjunction with eye gaze as a method of targeting. For example, when the user is looking at a button, the button can be activated by saying "Open." We provide the user feedback with the word "Open" showing up in front of the user for 2 seconds. We also allow users to directly say the names of sightseeing in the city model, such as "Times Square" to show the views of Times Square.

Visualization Results

As shown in Figure 3, we describe the architecture of our system with several visualization examples. The main menu consists of three sections: the "Base Maps" section, the "Info Layers" section, and the "Sightseeing" section. In our scenario, a

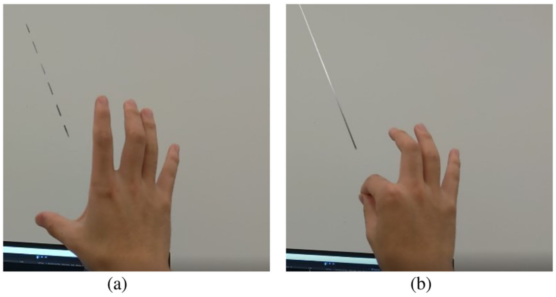


Figure 4. Far interaction of hand tracking is composed of two stages: the pointing stage to indicate the target object and the committing stage to manipulate the object. (a) Pointing stage. (b) Committing stage.

base map is used to provide a background of the geographic context of our city model. We provide three types of base maps including topography base maps [see Figure 3(a)], imagery base maps [see Figure 3(b)], and streets base maps [see Figure 3(c)]. The topographic base map provides a detailed and accurate representation of the elements on the surface of the earth. An imagery base map includes satellite imagery for the world, along with high-resolution aerial imagery for many areas, whereas a streets base map shows road and transit network details that are legible and accurate. The "Info Layers" section provides geographical information that can be displayed as tiles on the base map. We divide all the info layers into three aspects: environment, infrastructure, and people. Users have a variety of options to choose from in each aspect. For example, we provide the industrial area [see Figure 3(d)], population density [see Figure 3(e)] , and poverty distribution in New York City in the "People" related layers and subways [see Figure 3(f)], buildings, and transit frequency situation in the "Infrastructure" related layers. In our "Sightseeing" section, a few of well-known attractions are displayed as options, such as Times Square [see Figure 3(g)], Statue of Liberty [see Figure 3(h)], Brooklyn Bridge [see Figure 3(i)], and Central Park. When the base map with info layers or one of the attractions is selected, users will be able to see the detailed views.

QoE Metric Design

Modeling the QoE of XR systems has been extensively researched.[8] However, these QoE models are primarily designed for evaluating performances where frames are locally rendered by XR devices. In remote rendering XR systems, users are highly sensitive to latency, due to the delay introduced by video encoding, network delay, and video decoding, which are necessary steps for real-time rendered video streaming from the server. Unfortunately, few research studies have been conducted on QoE metrics for remotely rendered XR systems. In this article, we propose a time window-based QoE model for this purpose, which is defined as

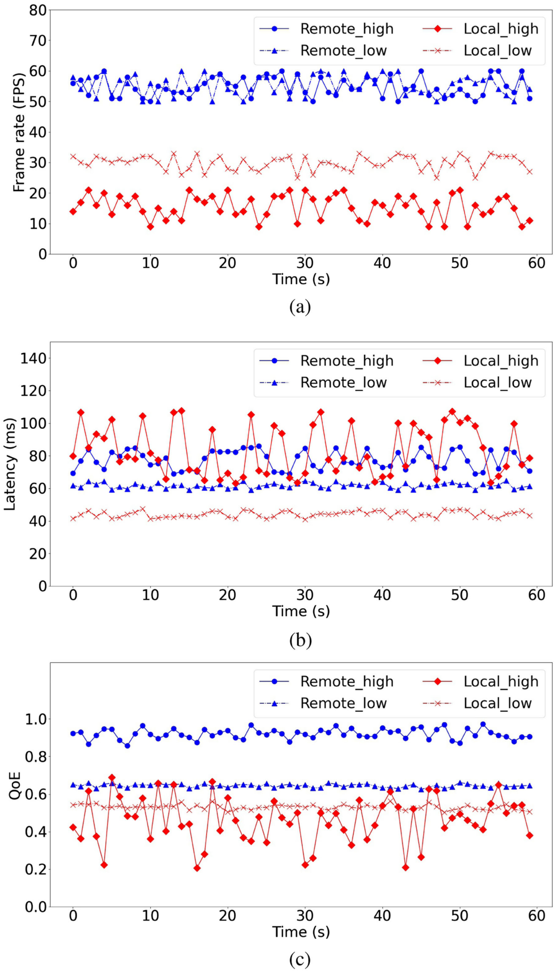
$$\mathrm{QoE} = \sum_{n=1}^{N} q(F_n) + \sum_{n=1}^{N} p(R_n) - u \sum_{n=1}^{N} g(L_n) \qquad (1)$$

where $q(F_n)$ represents the level of a user's satisfaction with the smoothness of the rendered video, where F_n donates the average frame rate at time window n. The level of satisfaction with video quality is indicated by $p(R_n)$, where R_n represents the average resolution at time window n. $g(L_n)$ is used to penalize the turnaround latency between sending pose data from XR devices to the remote server and displaying the video frame for that pose data on the display, where L_n donates the average latency at time window n. Because the marginal improvement in perceived quality decreases with higher frame rates and resolutions,[11] we used two logarithmic functions to represent $q(F_n)$ and $p(R_n)$, where $q(F_n) = \log(F_n/F_{\min})$ and $p(R_n) = \log(R_n/R_{\min})$. $F_{\min}$ and $R_{\min}$ are the minimum values of the frame rates and resolutions. Conversely, a user's satisfaction significantly decreases more as the total latency increases. Therefore, we used an exponential function to denote $g(L_n)$, i.e., $g(L_n) = e^{L_n/L_{\min}}$ and u is the latency penalty factor.

Remote Rendering versus Local Rendering

We evaluated our proposed remote rendering enabled XR system by comparing its performances with local rendering by the HoloLens 2 itself. For both cases (remote rendering and local rendering), the system is required to achieve 60 FPS with a resolution of 2048x1080 and a latency of approximately 60 milliseconds (ms) to provide a satisfactory experience.[12] In our experiments, we compared the performances of the systems in both cases using the abovementioned parameters as the target frame rate and the target resolution. We also evaluated the case when the target resolution is 1024x540 to further investigate their performances in high-quality resolutions and low-quality resolutions, respectively.

Four scenarios are tested and compared in terms of frame rate, latency, and QoE (as shown in Figure 5), including the remotely rendered high-quality videos (denoted as Remote_high), the remotely rendered low-quality videos (Remote_low), the locally rendered high-quality videos (Local_high), and the locally rendered low-quality videos (Local_low). We observe that the frame rates are far below the required level (60 FPS) when rendering the New York City model locally.



(a)

(b)

(c)

Figure 5. Comparison of our proposed remotely rendered system with the locally rendered system in terms of (a) frame rate, (b) latency, and (c) QoE for high-quality resolution (2048 × 1080) and low-quality resolution (1024 × 540). The QoE calculation equation is presented in Equation 1.

For the Local_high videos, the frame rates are scattered from 9 to 21 FPS, while for the Local_low videos, the frame rate is a little better, but still quite low (around 25–33 FPS). In contrast, the frame rates of remote rendering are stable and maintain around 55–60 for both the Remote_high and Remote_low videos. The reason might be that a considerable amount of time is spent on rendering frames by the GPU on the HoloLens 2 compared to the remote rendering. The smoothness of the rendered videos is remarkably improved on the powerful rendering server. Regarding latency, the Local_low videos perform the best (around 40 ms). However, for the Local_high videos, the latency can be as high as 110 ms, which may lead to sickness. Although remotely rendered systems introduce extra latency (such as encoding latency, transmission latency, and decoding delay), the total latency is much lower than the Local_high videos: 59–65 ms for the Remote_low videos and 68–86 ms for the Remote_high videos. In regard to QoE defined in Equation 1, Remote_high outperforms the other three scenarios at each time window, with significant improvement in the average QoE compared to the Local_high (0.92 versus 0.46). Remote_low also improves the average QoE by 21% compared to the Local_low (0.64 vs 0.53). Due to the limited computing resources on the HoloLens 2, the QoE scores for the Local_high videos are quite low and unstable.

NETWORK TRAFFIC ANALYSIS OF THE PROPOSED XR SYSTEM

XR Network

XR network[13] is defined as a network that carries XR contents. Specifically, in our system, the XR network connects the HoloLens 2 and the rendering server. To analyze the characteristics of the XR network in our system, we ran Wireshark, a packet sniffer, on the rendering server to capture the network traffic. The traffic analysis was performed at the default frame rate of the HoloLens 2 (60 FPS), with the default resolution of the HoloLens 2 (2048 × 1080) high-efficiency video coding as the compression standard and no limitation on the data rate. The captured network packets were stored in our local databases. We decoded the packets by Scapy, a Python library for manipulating network packets, to acquire key information, such as protocols in use and packet sizes. We found in our system that user datagram protocol (UDP) sockets over IPv4 are used instead of transmission control protocol (TCP). The reason could be that UDP is faster, simpler, and more efficient than TCP for XR systems where latency has a higher priority than a small amount of data loss.

XR Network Traffic Modeling

To design an effective and high-performance XR network, it is necessary to accurately characterize and model the traffic of XR networks. Modeling XR network traffic can also be highly useful for simulation studies, and generation of synthetic XR

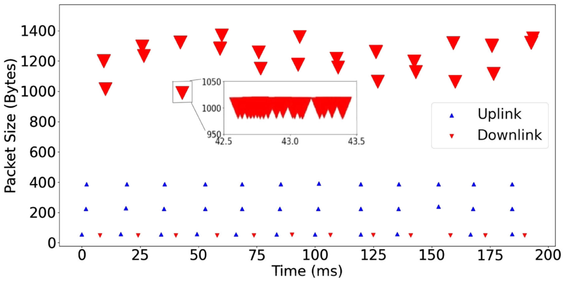


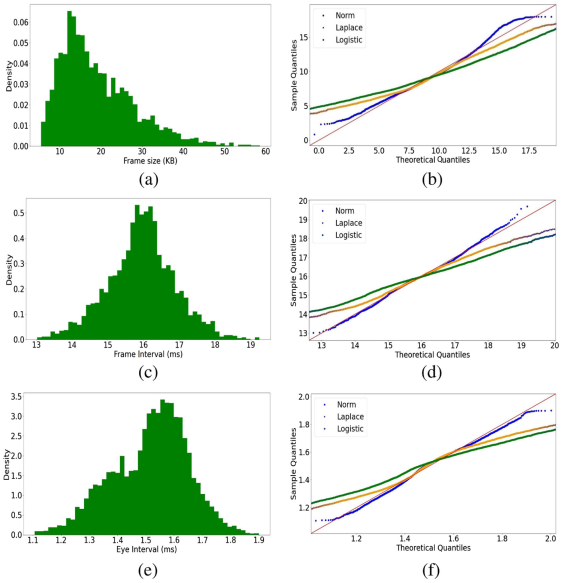
Figure 6. Part of the captured packets from the UL and DL stream in the XR network. A video frame is transmitted by two packet bursts for two eyes. Each burst contains 8–55 long packets with more than 1000 Bytes.

traffic traces for testing.[13] Among the various characteristics of XR networks, the following two are of particular interest: the distributions of video frame data and video frame size prediction.

Distributions of Video Frame Data We illustrate a portion of the bidirectional network traffic generated by our system, as shown in Figure 6. In the UL stream, we can see that two or three packets are sent almost simultaneously from the HoloLens 2 every 17 ms, which is similar to the time interval between two frames. Because the frame rate is 60 FPS, the expected time for a frame is 16.7 ms. We assume that these packets are mainly used as user data and synchronization information for our system to render frames. For the DL stream that contains rendered frames and synchronization information, there are two types of packets with different sizes. Long packets with more than 1000 Bytes are sent in bursts of multiple packets and every 17 ms two adjacent bursts are sent. We believe this is because the rendering server sends back the rendered frames for two eyes by two close bursts every 17 ms. Short packets containing synchronization messages from our system are sent separately at the same interval. Therefore, the frequency of data transmission between the HoloLens 2 and the remote system depends on the number of frames displayed on the HoloLens 2 per second.

We focused on the distributions of the video frame size, frame arrival interval, and arrival interval between two bursts for two eyes (the eye interval). As shown in Figure 7(a), we demonstrate the distribution of the video frame size using histograms. We constructed a quantile-quantile (Q-Q) plot where the sample points fall approximately on the line $y = x$ if the two distributions being compared are similar.[13] To see how the sample data fits different distributions, we compared the sample data to the normal distribution $(\mathrm{Normal}(x) = e^{-\frac{x^2}{2}}/\sqrt{2\pi})$, the Laplace distribution $(\mathrm{Laplace}(x) = e^{-|x|}/2)$, and the logistic distribution $(\mathrm{Logistic}(x) = e^{-x}/(1 + e^{-x})^2)$, with probability density functions in their standardized forms.

As shown in Figure 7(b), the sample data points for the normal distribution almost lie on the line $y = x$, indicating that the frame size can be considered to be subject to the normal distribution. Moreover, the sample data fits better on a normal distribution compared to the other two distributions as the line for the normal distribution lies closer to the line $y = x$. We can get the similar conclusions for the distributions of the frame interval [shown in Figure 7(c) and (d)] and the eye interval [shown in Figure 7(e) and (f)].



(a) (b) (c) (d) (e) (f)

Figure 7. Distributions of the video frame size, frame interval, and eye interval. We employed the Q-Q plots where we compared our sample data to the normal distribution, Laplace distribution, and logistic distribution. (a) Distribution of frame size. (b) Q-Q plot of frame size. (c) Distribution of frame interval. (d) Q-Q plot of frame interval. (e) Distribution of eye interval. (f) Q-Q plot of eye interval.

Therefore, we summarize that the frame size, frame interval, and eye interval of our system are all subject to the normal distribution.

Video Frame Size Prediction Predicting the oncoming video frame sizes is challenging because XR network traces exhibit both long-range dependent and short-range dependent properties. Several models have been proposed for traditional hypertext transfer protocol (HTTP)-based video traffic in the literature.[14] However, XR network traffic typically uses the real-time transport protocol instead of the HTTP as its protocol and displays several different characteristics, such as elephant DL stream and mice UL stream compared with the traditional video traffic. To the best of our knowledge, there is little work on XR traffic prediction. We denoted the video frame size by a time series of data F_t and proposed an autoregressive moving average (ARMA) model for predicting future values in the series. The model is referred as the ARMA(*p*, *q*) model where *p* is the order of the autoregressive (AR) part and *q* is the order of the moving average (MA) part, defined as

$$F_t = c + \epsilon_t + \sum_{i=1}^{p} \phi_i F_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \qquad (2)$$

where $\phi_i, \ldots, \phi_p$ are the parameters of the AR model, $\theta_i, \ldots, \theta_q$ are the parameters of the MA model, $\epsilon_t, \ldots, \epsilon_{t-q}$ are white noise error terms, and c is a constant. Because a stationary time series whose properties do not depend on time is easier to predict by an ARMA model, we first checked the stationarity of the video frame size series using the augmented Dickey–Fuller test. The results indicated that the series is stable with a 95% confidence interval. We then drew an autocorrelation factor plot and a partial autocorrelation factor plot to determine the parameters of the ARMA model (based on our analysis, *p* is 5 and *q* is 4). In our experiment, 70% of the data was used to train the model and the remaining 30% was used to evaluate its accuracy. We observe that the differences between the predicted values and the real values in the testing are not big (as shown in Figure 8), which verifies that the proposed model captures the autocorrelation structure of the frame sizes even when dramatic fluctuations in the frame size occur (between 4–6 s).

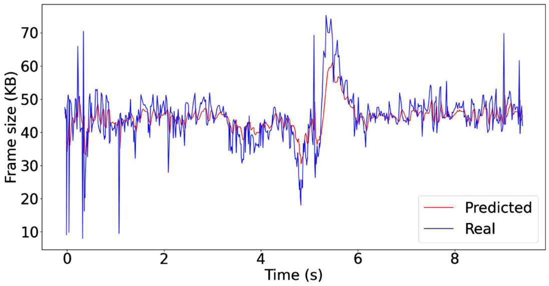


Figure 8. Predicted frame sizes by our ARMA-based model compared with the ground truth of the frame sizes on the test dataset.

Discussion and Future Work

There are still other challenges that we have not solved yet. It is noted that latency plays an important role in XR systems. The round-trip latency should be less than 20 ms for the motion-to-photon latency to become imperceptible.[15] If the latency is too large, users may experience the symptoms of motion sickness.[16] However, using a remote rendering server introduces extra latency for encoding frames, transmitting encoded frames by networks, and decoding frames. Another important challenge of remote rendering on the HoloLens 2 is the throughput of networks for transmitting high-quality video frames. For the HoloLens 2 with a frame rate of 60 Hz and a resolution of 2048×1080, if 24 bits are used for a single pixel, the raw data rate would be 2.96 Gbps.[12] Therefore, the future work is to figure out a way to lower the latency and increase the capacity for our XR network.

CONCLUSION

In this article, we demonstrated a remote rendering system for interacting high-quality holographic city data with the HoloLens 2 in real time. To verify the performance of remote rendering, we evaluated the proposed system by comparing its performance with local rendering solely conducted by the HoloLens 2 in terms of frame rate, latency, and QoE. The results show that the system is capable to fulfill smooth remote rendering transmission. In addition, the traffic characteristics of the proposed XR network, such as the frame size, frame interval, and eye interval were analyzed and modeled, which would benefit the design of any remote rendering systems based on the OpenXR standard.

REFERENCES

1. Å. Fast-Berglund, L. Gong, and D. Li, “Testing and validating extended reality (XR) technologies in manufacturing,” *Procedia Manuf.*, vol. 25, pp. 31–38, 2018, doi: 10.1016/j.promfg.2018.06.054.
2. P. Milgram and F. Kishino, “A taxonomy of mixed reality visual displays,” *IEICE Trans. Inf. Syst.*, vol. 77, no. 12, pp. 1321–1329, 1994.
3. P. Rosedale, “Virtual reality: The next disruptor: A new kind of worldwide communication,” *IEEE Consum. Electron. Mag.*, vol. 6, no. 1, pp. 48–50, Jan. 2017, doi: 10.1109/mce.2016.2614416.
4. Y. Yuan, “Paving the road for virtual and augmented reality [Standards] ,” *IEEE Consum. Electron. Mag.*, vol. 7, no. 1, pp. 117–128, Jan. 2018, doi: 10.1109/mce.2017.2755338.
5. L. Zhang, S. Chen, H. Dong, and A. El Saddik, “Visualizing Toronto city data with HoloLens: Using augmented reality for a city model,” *IEEE Consum. Electron. Mag.*, vol. 7, no. 3, pp. 73–80, May 2018, doi: 10.1109/mce.2018.2797658.
6. T. Alsop, “Augmented (AR), virtual reality (VR), and mixed reality (MR) market size worldwide from 2021 to 2024,” 2021. Accessed: Apr. 5, 2022. [Online]. Available: https://www.statista.com/statistics/591181/global-augmented-virtual-reality-market-size/
7. Y. Liu, H. Dong, L. Zhang, and A. El Saddik, “Technical evaluation of HoloLens for multimedia: A first look,” *IEEE MultiMedia*, vol. 25, no. 4, pp. 8–18, Oct.–Dec. 2018, doi: 10.1109/mmul.2018.2873473.
8. L. Zhang, H. Dong, and A. E. Saddik, “Towards a QoE model to evaluate holographic augmented reality devices,” *IEEE MultiMedia*, vol. 26, no. 2, pp. 21–32, Apr.–Jun. 2019, doi: 10.1109/mmul.2018.2873843.
9. “ArcGIS living atlas of the world.” Accessed: Apr. 5, 2022. [Online]. Available: https://livingatlas.arcgis.com/
10. M. Funk, M. Kritzler, and F. Michahelles, “HoloLens is more than air tap: Natural and intuitive interaction with holograms,” in *Proc. 7th Int. Conf. Internet Things*, 2017, pp. 1–2, doi: 10.1145/3131542.3140267.
11. K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, “BOLA: Near-optimal bitrate adaptation for online videos,” *IEEE/ACM Trans. Netw.*, vol. 28, no. 4, pp. 1698–1711, Aug. 2020, doi: 10.1109/infocom.2016.7524428.
12. L. Liu *et al.*, “Cutting the cord: Designing a high-quality untethered VR system with low latency remote rendering,” in *Proc. 16th Annu. Int. Conf. Mobile Systems, Appl., Serv.*, 2018, pp. 68–80, doi: 10.1145/3210240.3210313.
13. M. Lecci, M. Drago, A. Zanella, and M. Zorzi, “An open framework for analyzing and modeling XR network traffic,” *IEEE Access*, vol. 9, pp. 129782–129795, 2021, doi: 10.1109/access.2021.3113162.
14. S. Tanwir and H. Perros, “A survey of VBR video traffic models,” *IEEE Commun. Surv. Tut.*, vol. 15, no. 4, pp. 1778–1802, Oct.–Dec. 2013, doi: 10.1109/SURV.2013.010413.00071.
15. R. Ju *et al.*, “Ultra wide view based panoramic VR streaming,” in *Proc. Workshop Virtual Reality Augmented Reality Netw.*, 2017, pp. 19–23, doi: 10.1145/3097895.3097899.
16. E. Chang, H. T. Kim, and B. Yoo, “Virtual reality sickness: A review of causes and measurements,” *Int. J. Hum.-Comput. Interact.*, vol. 36, no. 17, pp. 1658–1682, 2020, doi: 10.1080/10447318.2020.1778351.

Zijian Long is a Ph.D. candidate with the School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada. His research interests include XR network and artificial intelligence. Long received his M.Sc. degree in electrical and computer engineering from the University of Ottawa. Contact him at zlong038@uottawa.ca.

Haiwei Dong is a principal researcher with Waterloo Research Center, Huawei Technologies Canada, Waterloo, ON, Canada, and a registered Professional Engineer in Ontario. He was a principal engineer with Artificial Intelligence Competency Center, Huawei Technologies Canada, a research scientist with the University of Ottawa, Ottawa, ON, Canada, a postdoctoral fellow with New York University, New York City, NY, USA, a research associate with the University of Toronto, Toronto, ON, Canada, and a research fellow (PD) with the Japan Society for the Promotion of Science, Tokyo, Japan. His research interests include artificial intelligence, multimedia communication, multimedia computing, and robotics. Dong received his Ph.D. degree in computer science and systems engineering from Kobe University, Kobe, Japan. Contact him at haiwei.dong@ieee.org.

Abdulmotaleb El Saddik is a distinguished professor with the School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada, and a Professor with the Mohamed bin Zayed University of Artificial Intelligence. He has supervised more than 120 researchers. He has coauthored ten books and more than 550 publications and chaired more than 50 conferences and workshops. His research interests include the establishment of digital twins to facilitate the well-being of citizens using AI, IoT, AR/VR, and 5G to allow people to interact in real time with one another as well as with their smart digital representations. He is a fellow of Royal Society of Canada, a fellow of IEEE, and an ACM distinguished scientist and a fellow of the Engineering Institute of Canada and the Canadian Academy of Engineers. Contact him at elsaddik@uottawa.ca.