Improvement of Service Qualities by Edge Computing in Network-oriented Mixed Reality Application

Shiori Takagi*, Junichi Kaneda*, Shin'ichi Arakawa* and Masayuki Murata*
*Graduate School of Information Science and Technology, Osaka University Yamadaoka 1–5, Suita, Osaka, 565–0871 Japan
{s-takagi, j-kaneda, arakawa, murata}@ist.osaka-u.ac.jp

***Abstract*—A key challenge for developing networked Cyber-Physical System is how to integrate and process the virtual and real-world information from locally or remotely connected humans/robotics with a tolerable application latency. In this paper, we investigate the improvement of application latency by introducing edge computing environments and the resulting service quality through some experiments. A network-oriented mixed-reality (MR) application that operates a remote robot through users' gestures and displays location-aware information through edge server is implemented as a simplified implementation of cyber-physical networking applications. The results of our experiments reveal that service quality suddenly gets worse when application latency becomes around 1 [sec]. In other words, the service quality of the network-oriented MR application is expected to be improved by introducing edge computing when latency of application running on cloud computing environments is about 1 [sec].**

***Index Terms*—Mixed Reality (MR), Edge Computing, Quality of Service (QoS), Network Robot, Network Application**

I. Introduction

With the rapid development of smart phones and tablets, networked services have become more familiar in our daily life. The information retrieved from cameras or sensors of smart phones is increased and diversified, which pursues new networked services such as Cyber-Physical Systems (CPS) and/or highly networked CPS, namely Cyber-Physical Networking.

CPS is considered, in a wide sense, as the system of interactions between virtual information and real-world information, and is expected to offer an sophisticated control of movements for humans/robotics in the real world. As a matter of course, the networked version of CPS should consult interactions among humans/robotics, which may be geographically distributed, through a communication channel.

A key challenge for developing the cyber-physical networking is how to integrate and process the virtual and real-world information from locally or remotely connected humans/robotics with a tolerable application latency. In recent years, research and development of edge computing (MEC: Multi-access Edge Computing) is progressing in the field of network research. In cloud computing, data centers process information acquired from cameras or sensors in remote place, and it takes hundreds of milliseconds because of distance and load concentration. In edge computing, computing resources and storage are allocated at the edge of the network, so that processing end devices require is performed at the place closer to the end devices. As shown in Fig. 1, it is expected that application responsiveness is improved by edge computing [1], [2].

In this paper, we conduct experiments to examine the improvement of latency by edge computing for applications of cyber-physical networking. For this purpose, we develop a network-oriented mixed-reality (MR) application that operates a remote robot through users' gestures as a simplified implementation of cyber-physical networking applications. Note that, recent trends of virtual reality (VR) technology and mixed reality (MR) technology have also led to the development of network services that provide a realistic experience. For example, a service that delivers wedding ceremonies to relatives wearing a VR headset in remote area [3] and a "super high presence public viewing," that provides remote spectators with experience to feeling as if athletes appear from the sports competition stadiums [4] are now being considered. As like these applications, our application transfers the information between local/remote devices and provides realistic experience by live streaming video. But, unlike these applications, our application integrate and process the real-world information of user and robot at the edge, thereby application latency will be reduced accordingly.

Application latency is always reduced by the edge computing (as long as we do not consider a congestion at edge servers). Thus, our focus of this paper is to examine the improvement of service quality rather than the improvement of application latency. In this paper, we perform a subjective evaluations, where users' perceived service quality is scored by MOS (Mean Opinion Score), and evaluate the improvement of the service quality by introducing edge computing. Because MOS-based evaluation has been introduced for evaluating the quality of streaming services [5]–[7], we newly introduce four perspectives, 1) comfort of robot operations, 2) immersion, 3) visibility on MR headset, 4) visibility on video streaming, to quantify the service quality of our MR applications.

The rest of this paper is organized as follows. Section 2 describes the application we developed. In Section 3, we explain the experiment environment, evaluation method, and experimental results. The conclusion of this paper is summarized in Section 4.

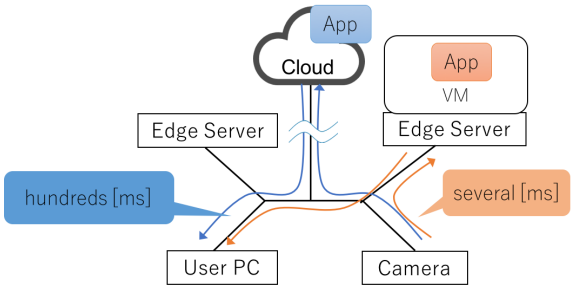


Fig. 1: An example of information processing with edge computing.

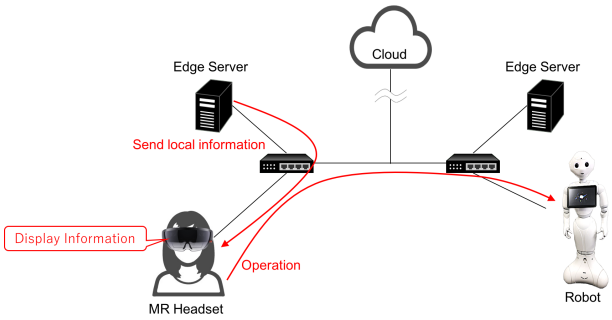


Fig. 2: User-side application.

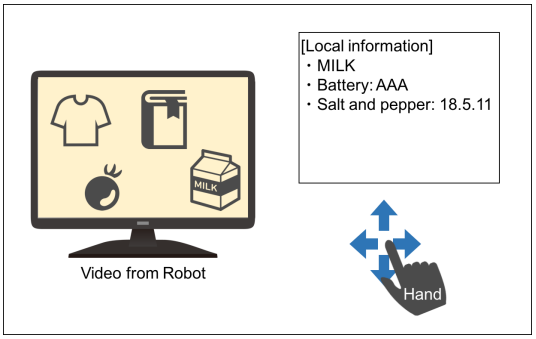


Fig. 3: Illustration of our application.

II. Network-oriented MR Application

In this section, we describe the network-oriented mixed reality application supposed in this paper II-A and the application we actually implemented II-B.

A. Supposed Service

The supposed service is a shopping mall experience service using mixed reality technology. This service offers a shopping experience at users' home by operating a robot placed at actual shopping stores.

The robot takes videos inside shops in the real world, inserts the local information such as stock of products, and best-selling products, and transfers these information to the user. Here, the local information is stored at the edge server of robot side (store side) and updated either periodically or occasionally. In addition, the application on the edge server inserts the local information at any time in accordance with the movement of the robot and transmits it to users. In the future, not only video and audio but also information of five senses such as odor and tactile sensation of the product may be transmitted to the user to provide more realistic experience.

Users at home wear MR headsets, and local information of the user side, such as stuffs that is out of stock, foods that has passed the freshness date, is displayed on the MR headsets. The local information of the user side is stored in the edge server close to the user's house, and the information is automatically updated by transmitting the captured video of MR headset to the edge server and video processing at the edge server.

A user operates the robots by gesture and voice by watching the video sent from the remote robot and the information displayed on the MR headset.

B. Implemented Application

The application program we implemented consists from three sub-programs: moves the remote robot by user's gesture, displays the information at the user side on the MR headset, overlays the information at the remote side on a video stream from the remote robot to the user. The first two sub-programs are implemented as user-side application, and the last sub-program is implemented as robot-side application. We explain the specifications of user-side application and robot-side application in following sections.

1) User Side: Fig. 2 shows the process in user-side application. Figure 3 shows the view through the MR headset when the user is running the application.

The object which shows local information is displayed on the MR headset when the application is launched. The local information is stored at the edge server of user side and is retrieved through HTTP access. The MR headset periodically polls the local information stored in the server by HTTP and displays its contents on the MR headset.

In addition, the MR headset detects the movement of users' fingers. The robot is operated by the users' swipe or tap operation. Users' gesture is fixed when the user presses a finger down, moves the finger in one of the upward, downward, rightward, and leftward directions, and then raises the finger up. When the amount of movement of the user's finger exceeds a threshold value, the robot moves to either of dictions that the finger moves. The distance the robot moves at once is set to six times larger than the actual distance that the user's finger moves. When users do tap or swipe operations that do not exceed the threshold, the robot rotates 90 degrees in clockwise direction.

We use Pepper developed by Softbank Robotics Corporation as a robot for the experiment. Pepper provides many APIs to create applications. In our application, movement of Pepper is realized by using ALmotion API. As for the MR headset, we use Microsoft HoloLens. HoloLens is a head mounted

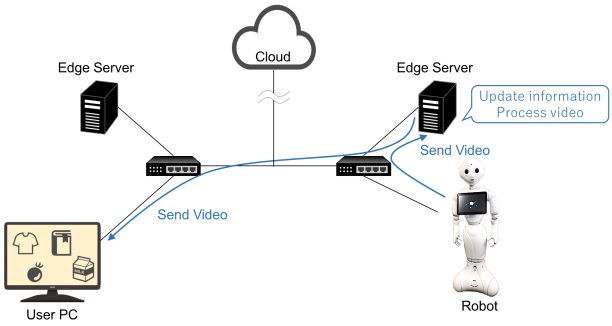


Fig. 4: Robot-side application.

wearable computer. HoloLens provides Holograms, which are virtual objects. Users can operate Hologram by gaze, gesture, and voice. HoloLens is equipped with a function to detect hand position and tapping operation. It calculates the movement distance of the hand and determines the movement distance of Pepper.

We developed the application by using Unity and C#. The object to display the local information is a Cube object of Unity, and the content of local information is retrieved by using WWW class of Unity.

HoloLens and Pepper are connected by a TCP connection.

2) Robot Side: Fig. 4 shows the process of the robot-side application.

At the robot side, the video taken by the robot is processed and delivered. We use FFmpeg [8], which is free software that can record and process images for acquisition, processing, and delivery of captured videos. Although FFmpeg 0.9.0 is installed in Pepper OS, NAOqi, the FFmpeg cannot stream videos over the network. Therefore, we recompiled FFmpeg 0.9.0 such that video streaming over the network is supported.

We also use FFserver, which is a streaming server attached to FFmpeg for streaming distribution of video, and FFplay, which is a player to show the video distributed by FFserver on the displays. Many parameters can be set to these software. Among them, we set the video buffer size to minimum value in order to suppress delay due to video buffering.

C. Execution of Application

Fig. 5 shows an actual view when our application is running. The area surrounded by blue color is the video from the robot-side and the area surrounded by red color is a virtual object displayed on the MR headset. Inside the virtual object, the stuffs that the user should buy are listed as the local information.

D. Edge Computing Environments

Since users and robots are supposed to be placed at geographically different places, edge servers are located on each of the user side and the robot side. The edge server on user side stores local information related to user's environment. Similarly, the edge server on the robot side stores local information of robot's environment. The edge server on the

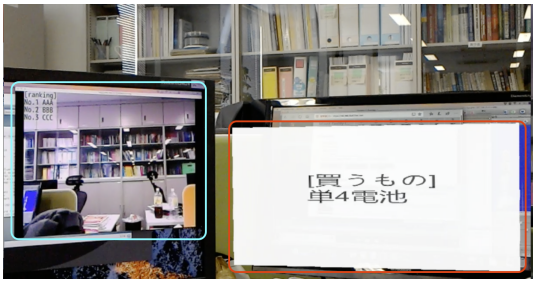


Fig. 5: Execution of application.

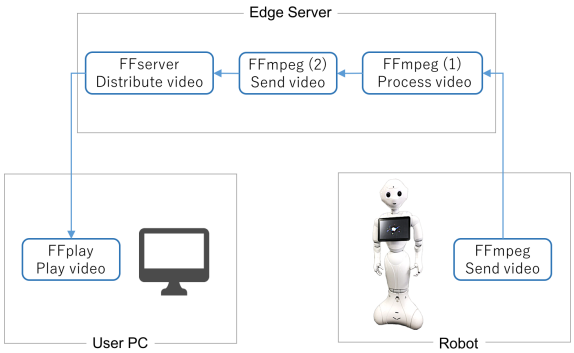


Fig. 6: The flow of video processing.

robot side also performs video processing and video streaming through FFmpeg and FFserver. Figure 6 shows the processing flow of video data in our application. FFmpeg running at the edge server of robot-side sends videos to FFmpeg (1) with UDP. FFmpeg (1) inserts a text into as an overlaid image into video received from the robot. FFmpeg (2) sends the video to FFserver using TCP. In our experiments, MPEG2 is selected as the video format because MPEG2 is streamable.

III. Evaluation of Service Quality

In this section, we describe the experiment environment, the method to evaluate service quality, and the result of the experiment.

A. Experiment Environment

Fig. 7 shows a network system constructed for our experiments. "SW" in the figure represents switches. We use OpenStack [9] to set up the network environment. Robot, MR headset, and user PC are connected to the wireless access points with IEEE 802.11n. In this experiment, we do not prepare the cloud computing environment. Instead, we generate a network delay at edge servers to emulate network delay that will occur in the cloud computing environment. The netem [10] is used to generate the network delay. Note that it takes about 420 [ms] for the robot to transfer a frame of video [11]. Thus, when we set x [ms] for netem, totally 420 + x [ms] is experienced by users.

In the experiment, we prepare four tasks shown in TABLE I, and measured the change in the task completion time when

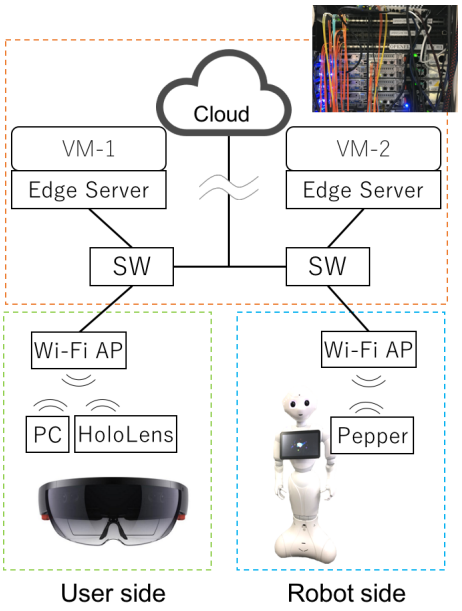


Fig. 7: Network structure for experiment.

TABLE I: Tasks and local information to refer in our experiments.

Task	Information to refer	Operation
1	none	move to the goal given beforehand
2	user-side	move to the goal shown in MR headset
3	robot-side	move to the goal shown in video
4	both	move to the goal shown in local information

the delay changed. In all tasks, participants move the robot to the goal. The goal of task1 is given beforehand, and goals of other tasks are displayed on the MR headset or the video after the operation started. When the marks of the goal appear in the video sent from the robot, it is assumed that the task is completed. At the first time, a participant operates the robot in the order of Task 1 to Task 4 with no network delay. Then, we manually changed the network delay to a specific value. After this, the participant again operates the robot in the order of Task 1 to Task 4 with network delay. As the preliminary experiment, one of authors measured the task completion time by setting the network delay to 0 [ms], 100 [ms], 200 [ms], 300 [ms], 400 [ms], and 500 [ms]. Finally, we conduct our experiments with eight participants. Four of them operate the robot with 0[ms] and 300 [ms] network delay, and the other four participants do the same tasks with 0 [ms] and 500 [ms] network delay.

B. Evaluation method

Mean Opinion Score (MOS) [12] is a numerical indication to evaluate quality of experience. Ref. [12] describes how to perform subject experiments and evaluate quality of experience. However, the method described in Ref. [12] is an evaluation method for video and audio quality, and the method to evaluate the quality of experience in the mixed reality applications has not been specified. Therefore, in this paper, we evaluate service quality by the following method.

- **Objective evaluation:** We measured task completion time under each condition, and compared task completion time.
- **Subjective evaluation:** Each of participants evaluates the quality of second operations (with 300 [ms] or 500 [ms] network delay) compared with first operations (with no network delay) with the categories shown in TABLE II for four perspectives (TABLE III) after finishing all tasks.

TABLE II: Categories and Scores for evaluation.

Category	Score
Much Better	3
Better	2
Slightly Better	1
About the Same	0
Slightly Worse	-1
Worse	-2
Much Worse	-3

TABLE III: Evaluation items.

E1	Quality of the video taken by the robot
E2	Comfort of robot operation
E3	Immersion
E4	Visibility of the information displayed on the MR headset
E5	Visibility of the information displayed on the video

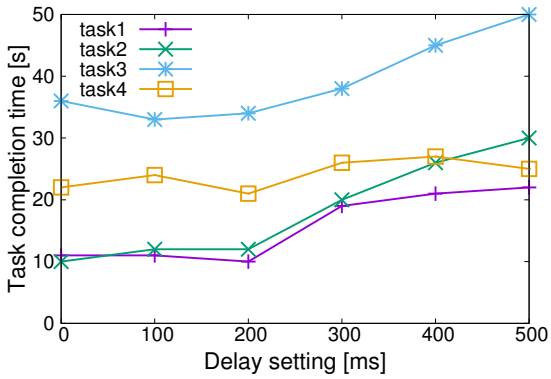


Fig. 8: Result of the preliminary experiment.

C. Result of the Experiment

We describe results and considerations of objective evaluation and subjective evaluation.

1) Objective Evaluation: Fig. 8 shows the result of our preliminary experiment. The horizontal axis represents the network delay and the vertical axis represents the task completion time. As supposed, the task completion time increases as the the network delay increases. However, the task completion time significantly increases for Task 1, Task 2, and Task 3 when network delay exceeds 300 [ms]. This suggests that the network delay of 300 [ms], or totally 720 [ms] application

latency, may be a threshold of impact on the task completion time. Only the task completion time of Task 4 did not increase significantly even when network delay increases. We believe that the participant experiences the network delay from Task 1 to Task 3, and thus can predict how much delay will occur on operating the robot.

Fig. 9 shows task completion time for each task with eight participants. From the figure, we observe that

- Completion time of Task 1 does not change significantly. Since Task 1 requires few operation, difference of network delay does not affect the task completion time so much. The task completion time of Task 2 and Task 3, which is more complicated task than Task 1, change more than Task 1.
- The task completion time of Task 3 is the largest among all tasks. This is because total distance and number of operations that the participant has to operate are largest among tasks.
- The task completion time of Task 4 do not change as much as Task 3. This is also observed in our preliminary experiments.

We also observe that, when network delay was 500 [ms], task completion time varied more widely than that when network delay was 0 [ms] and 300 [ms]. Some participants predicted the large network delay and perform multiple operations at once, while other participants checked the results of operations through a video every time they operated.

The fact that nonlinearity of the task completion time to network delay was not observed suggests that there was no puzzlement or operation error due to delay of the video. This is because tasks given to subjects were simple and subjects predicted the delay of the video when they operated the robot although we did not tell participants that we generated network delay. The influence of network delay in more complicated operations and robot operations with high-precision needs to be investigated in the future.

2) Subjective Evaluation: Fig. 10 shows the averaged value of MOS for participants. The horizontal axis represents evaluation perspectives and the vertical axis represents the MOS for two network delay settings.

The results of Fig. 10 suggests that,

- MOS of E1 gets worse in both conditions where the network delay was 300 [ms] and 500 [ms]. In our experiment, participants can feel the video quality during the movement of robot after the participants completed their gestures. Therefore, it is reasonable that MOS decreases as the increase of network delay.
- MOS of E2 gets better when the network delay was 300 [ms], while it gets worse sharply when the network delay was 500 [ms]. Participants perform the same task twice continuously under different conditions.
- MOS of E3 is same when network delay was 300 [ms], but it gets worse when network delay was 500 [ms]. This indicates that service quality regarding E3 drops suddenly when network delay exceeds a certain value between 300 [ms] and 500 [ms]. Since the processing delay in the robot is about 420 [ms], service quality drops suddenly when the application latency between users and robots is between 720 [ms] and 920 [ms]. In other word, service quality is expected to be improved by introducing edge computing when application latency reaches 1 [sec] in cloud computing environment.
- MOS of E4 and E5 does not change. In this experiment, MOS of E4 and E5 were not affected by network delay because local information displayed on the MR headset and video was static.

Although task completion time increases almost linearly with the network delay in the objective evaluation, the results of MOS for E2 and E3 reveals that service quality drops suddenly when application latency reaches 1 [sec]. This fact suggest that task completion time is not sensitive to network delay, but quality of experience is sensitive to network delay. Therefore, it is not easy for network providers or service providers to estimate service quality of their services only with objective indicators based on operation logs. In particular, when we consider the service function placement in the edge computing environment, deep understanding of the human experience mechanism is required.

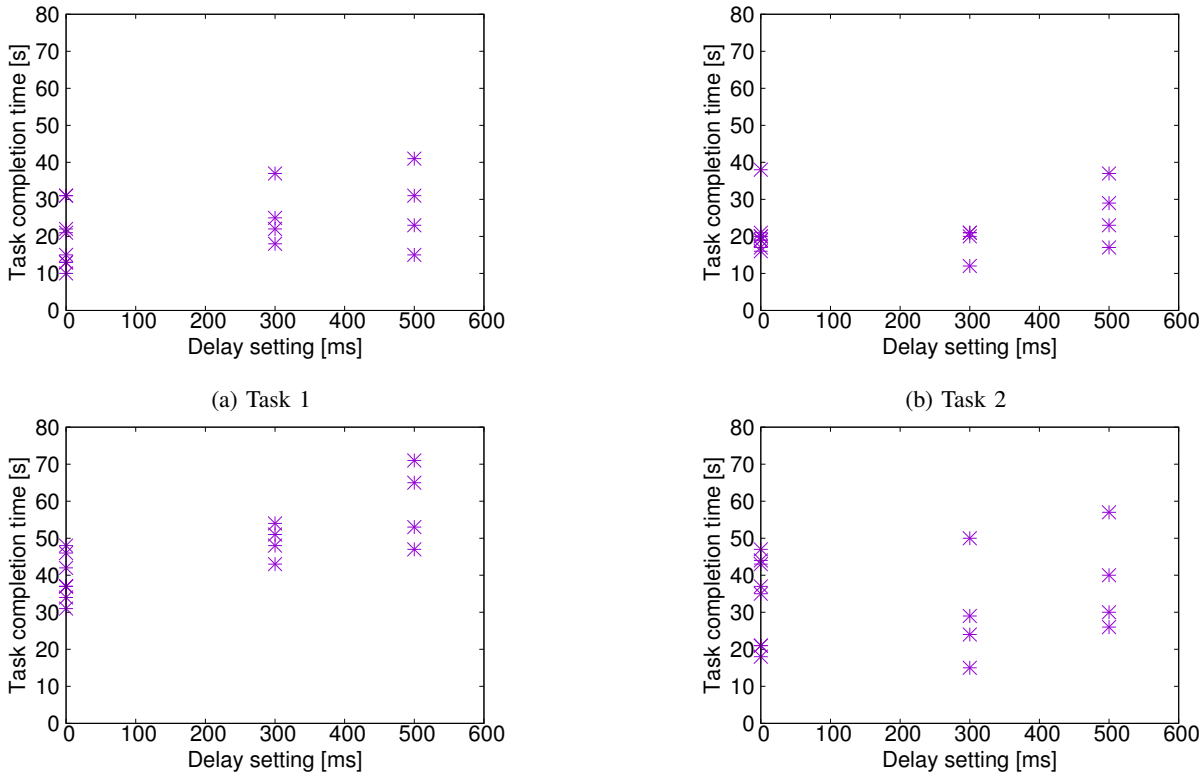
In our experiment, we always generate the same amount of delay at second time. In the actual network environment, the timing and amount of network delay are not constant. Performing experiments by randomly generating network delay while performing a series of users' operations makes the experiment more realistic and will measure the service quality more accurately.

IV. Concluding remarks

A key challenge for developing networked Cyber-Physical System is how to integrate and process the virtual and real-world information from locally or remotely connected humans/robotics with a tolerable application latency. In this paper, we investigated the improvement of application latency by introducing edge computing environments and the resulting service quality. For this purpose, we implemented a network-oriented mixed-reality (MR) application that operates a remote robot through users' gestures and displays location-aware information through edge server as a simplified implementation of cyber-physical networking applications.

Our objective evaluation and subjective evaluation reveal that quality of experience is sensitive to network delay. Therefore, service quality suddenly gets worse when application latency becomes around 1 [sec]. In other words, the service quality of the network-oriented MR application is expected to be improved by introducing edge computing when the application latency is about 1 [sec] in the cloud computing environments. The results also suggested that it is not easy for network providers or service providers to estimate service quality of their services only with objective indicators based on operation logs.

In our experiment, we always generate the same amount of delay at second time. In the actual network environment,



(a) Task 1

(b) Task 2

(c) Task 3

(d) Task 4

Fig. 9: Task completion time.

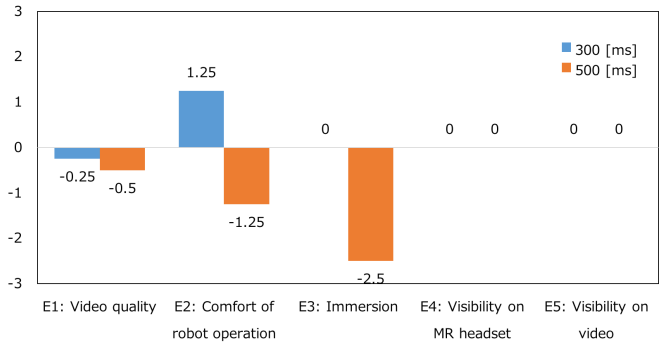


Fig. 10: Result of subjective evaluation.

the timing and amount of network delay are not constant. Performing experiments by randomly generating network delay while performing a series of users' operations makes the experiment more realistic and will measure the service quality more accurately, which will lead to understand the human experience mechanism in detail.

ACKNOWLEDGEMENT

Apart of this work was supported by National Institute of Information and Communications Technology (NICT) in Japan.

REFERENCES

[1] ETSI, "Mobile-edge Computing Introductory Technical White Paper," Sept. 2014.
[2] Hiroyuki Tanaka, Noriyuki Takahashi, and Ryuraro Kawamura, "Research and development of edge computing that opens the IoT era (in Japanese)," *NTT Technical Journal*, pp. 59–63, Aug. 2015.
[3] "HUG PROJECT." https://hugproject.net/.
[4] Ryuji Kubozono, Akihito Akutsu, Norihiko Matsuura, Kenichi Minami, and Akira Ono, "Beyond 2020: Creation of high reality UX service," *NTT Technical Journal*, pp. 6–9, Oct. 2017.
[5] I. Busse, B. Deffner, and H. Schulzrinne, "Dynamic QoS control of multimedia applications based on RTP," *Computer Communications*, vol. 19, pp. 49–58, Jan. 1996.
[6] International Telecommunication Union, "Recommendation ITU-T P.10/G.100, Vocabulary for performance, quality of service and quality of experience."
[7] Hideyuki Koto, Norihiro Fukumoto, Sumaru Niida, Hidetoshi Yokota, Shin'ichi Arakawa, and Masayuki Murata, "Users' reaction to network quality during web browsing on smartphones," *26th International Teletraffic Congress (ITC 26)*, Sept. 2014.
[8] "FFmpeg." https://www.ffmpeg.org/.
[9] "Openstack." https://www.openstack.org/.
[10] "netem." https://wiki.linuxfoundation.org/networking/netem.
[11] Junichi Kaneda, Shin' ichi Arakawa, and Masayuki Murata, "Effects of Service Function Relocation on Application-level Delay in Multi-access Edge Computing," *in Proceedings of IEEE 5G World Forum*, July 2018.
[12] International Telecommunication Union, "Recommendation ITU-T P.800, Methods for subjective determination of transmission quality."