Zero Interruption-Oriented Techniques for Mobile Video-on-Demand in Hybrid Broadcasting Environments

Tomoki Yoshihisa*, Tomoya Kawakami**, and Yusuke Gotoh***

* Cybermedia Center, Osaka University, Japan
** Graduate School of Information Science, Nara Institute of Science and Technology, Japan
*** Graduate School of Natural Science and Technology, Okayama University, Japan
yoshihisa@cmc.osaka-u.ac.jp, kawakami@is.naist.jp, gotoh@cs.okayama-u.ac.jp

Abstract - Due to the recent proliferation of mobile devices and video-on-demand delivery, mobile video-on-demand delivery, i.e., the users watch videos using mobile devices, gets great attention. In mobile video-on-demand delivery, the users often watch videos while moving outside such as riding cars or trains. Current mobile video-on-demand delivery faces the problem that interruptions of video playback occur in some situations. So, some interruption reduction techniques have been studied. However, these techniques are originally designed for non-mobile devices and it is difficult to solve the problem for mobile devices under various situations. Hence, in this paper, we propose some techniques aiming to zero interruption. To reduce the video interruptions effectively, our proposed techniques use hybrid broadcasting environments. We develop a mobile video-on-demand system with our proposed techniques and report our experiments of mobile video-on-demand delivery using the developed system[1].

Keywords: Internet Broadcasting, Continuous Media, Streaming Delivery, Mobile Devices

1 INTRODUCTION

Due to the recent development of Information and Communication Technologies (ICT), mobile devices such as smart phones or compact PCs become popular. Mobile devices are small, lightweight, and can connect to the Internet at most places. So, the users can get information from the Internet using mobile devices at most outside places even though they do not seat in front of non-mobile devices such as desktop PCs. Meanwhile, video-on-demand delivery such as Internet broadcasting services by YouTube or TV companies becomes popular due to the communication speed up of the Internet. In video-on-demand delivery, the users select their preferable videos from homepages and request the video deliveries to the delivery server. The delivery server sends the video data to the requested clients according to their requests. The users can watch the video from the playing position of the received data without waiting for the reception of all data by using streaming technique, in which the clients play the video data while receiving them. The proliferation of mobile devices and video-on-demand delivery leads mobile video-on-demand delivery, i.e., the users watch videos using mobile devices. In mobile video-on-demand delivery, the users often watch videos while moving outside such as riding cars or trains since they can select and watch videos at most places.

Current mobile video-on-demand delivery faces the following problems.

Problem 1: Interruptions occur when there are a large number of clients.

Interruptions of video playback occur when there are a large number of clients since the bandwidth between the server and each client decreases. For example, suppose the case when a user requests playing a popular video delivered in YouTube when he/she is stopping at a red traffic signal. In this case, the video playback sometimes does not start or interrupts even if it starts. This problem also occurs for non-mobile devices, but frequently occurs for mobile devices since there are a large number of mobile devices.

Problem 2: Interruptions occur when the condition of electric wave gets worse.

Mobile video-on-demand delivery is often used while moving and the condition of the electric wave for the Internet connection changes. A worse electric wave condition causes a lower bandwidth. So, the video playback interrupts when the bandwidth becomes lower than the bit rate. For example, a user requests playing a video while riding on a train at a station and starts watching it. When the train moves to a far distance from the station, the electric wave does not reach to his/her mobile device and the video playback interrupts.

Problem 3: Video playback stops when the remaining battery is low.

Mobile devices are often used outside and it is difficult to charge their batteries outside. So, the users may reduce the battery consumptions caused by video playbacks to lengthen the running time of the devices. The battery consumptions for playing high bit rate videos are high since the processing data size per time increases. Hence, in mobile video-on-demand delivery, some users prefer playing low bit rate videos to high bit rate videos so as to lengthen the time to play videos.

[1] This research was supported in part by Grant-in-Aids for Scientific Research (B) numbered 15H02702, and Grant-in-Aids for Challenging Exploratory Research numbered 26540045.

Some interruption reduction techniques have been studied. But, these techniques are originally designed for non-mobile devices and it is difficult to solve these problems. A simple approach to solve the problem 1 is exploiting streaming delivery techniques such as CDN (Contents Delivery Network) or P2P streaming. However, it is difficult to solve the problem by only these techniques when there are too many mobile devices, e.g., many mobile devices in Japan play the videos of disaster situations. Regarding the problem 2, mobile devices can continue to play videos even when they lack Internet connections by sufficiently buffering the video data. However, mobile devices cannot sometimes get enough time to buffer the data, e.g., the train moves fast or the power of the electric wave is weak. Regarding the problem 3, battery consumptions can be reduced by stopping other applications running on mobile devices. However, in this simple approach, the convenience of mobile devices degrades.

Hence, in this paper, we propose three techniques aiming to zero interruption. To reduce the interruptions of playing videos effectively, our proposed techniques use hybrid broadcasting environments. In hybrid broadcasting environments, the clients can receive data both from the broadcasting system and the communication system. The broadcasting system can deliver data to all clients concurrently. So, by broadcasting the data that are requested by many clients, the interruptions of playing videos are reduced effectively. Also, we develop a mobile video-on-demand system using our proposed techniques and report the experiments of mobile video-on-demand delivery using our developed system.

2 RELATED WORK

2.1 Types of Video-on-Demand Delivery

Video delivery through the Internet is called video-on-demand delivery. In video-on-demand delivery, the users select their preferable videos from the lists shown on homepages, etc. and request playing the video. The delivery server can deliver different videos to some clients (mobile devices). But, the interruptions of videos occur when there are a large number of the clients and the bandwidth between the delivery server and each client becomes smaller than the video bit rate. Hence, for communication systems such as the Internet, P2P streaming delivery techniques, in which clients also deliver videos to other clients, have been studied [1, 2].

Video delivery through broadcasting systems such as terrestrial broadcasting or one-segment broadcasting is called near video-on-demand delivery. Near video-on-demand delivery realizes short waiting time for starting playing videos by broadcasting data to all clients. In near video-on-demand delivery, some methods to reduce the interruption time by creating effective broadcast schedules of video data have been proposed [3, 4]. These methods divide the data into some segments and reduces the interruption time by broadcasting the segments repeatedly according to the created broadcast schedule. However, the interruption time of near video-on-demand delivery is longer than that of video-on-demand delivery when the number of the clients is small.

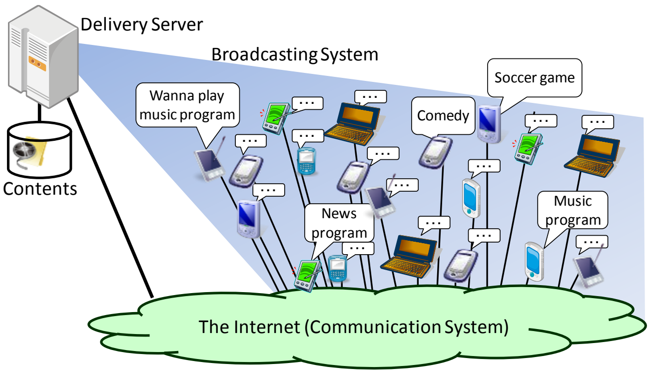

Figure 1: A hybrid broadcasting environment

Video delivery combining the video-on-demand and the near video-on-demand techniques is called unified video-on-demand delivery. Some methods for unified video-on-demand delivery have been proposed [5, 6]. In these methods, the delivery server fixes the broadcast schedule and sends the data, of that time to broadcast is long after, via the Internet. A method that generates the broadcast schedule dynamically is proposed in [7].

2.2 Video-on-Demand Services

Some video-on-demand systems have been developed for services.

One of the famous systems for delivering the videos that are submitted by the users is YouTube [8]. YouTube service started on 2004 at United States and widely used for delivering the users submitted videos. USTREAM also uses a video-on-demand system for delivering the users submitted videos [9]. USTREAM is often used for live broadcasting on the Internet. Niconico-douga is a Japanese video-on-demand service for users submitted videos [10].

Netflix provides a video-on-demand service for delivering the movies in Blu-rays or DVDs [11]. Netflix originally started a DVD rental service on 1997. Using the DVDs for the rental service, Netflix started the video-on-demand service. Hulu provides a similar video-on-demand service with charge [12]. There are many other video-on-demand services all over the world [13].

Techniques used in the above methods and services are originally designed for non-mobile devices and it is difficult to solve the problems for mobile devices under various situations.

2.3 Contribution of Our Work

Our proposed techniques for video delivery (the stream merge technique and the spare data delivery technique) are classified into unified video-on-demand delivery. Different from the delivery server of pure video-on-demand delivery or near video-on-demand delivery, the delivery server of unified video-on-demand delivery can deliver the video data both via the broadcasting system and the communication system. So, the video interruptions can be further reduced. Previously proposed methods for unified video-on-demand delivery focus on how to deliver the segmented video data to further reduce the interruptions. Our proposed techniques, moreover, focus on spare data and remaining batteries to reduce the interruptions. These techniques to reduce the interruptions are differences between our research and [5]-[7]. Also, different from existing video-on-demand services, we adopt hybrid broadcasting environments in the paper.

The overhearing function in Wi-Fi communications can be regarded as a broadcasting system for hybrid broadcasting environments and our proposed stream merge technique can be applied for the overhearing function. However, different from the series of pseudo-broadcast methods using the function, our proposed techniques include the spare data delivery technique and the remaining battery based bit rate technique. We explain the detail of these techniques in the next section.

3 PROPOSED TECHNIQUES

In this section, we explain our proposed techniques. First, we explain our assumed hybrid broadcasting environments. After that, we explain our proposed techniques to solve the problems described in Section 1.

3.1 Hybrid Broadcasting Environments

Figure 1 shows our assumed hybrid broadcasting environment. The clients in the broadcasting area can receive data from the broadcasting system. Also, they can request their preferable data to the server and can receive them from the communication system. The broadcast station delivers data via some broadcast channels and is managed by the server. The server has streaming data and can broadcast the data to the clients using the broadcast station. Also, it can send the data to the clients using the communication system by unicasting.

3.2 Stream Merge

To solve the first problem, we propose a mobile video-on-demand system using stream merge technique. When there are many mobile devices that receive video data, the video-on-demand system can avoid decreasing the bandwidth between the server and the clients by sending the data to some clients concurrently and reducing the communication traffic. For example, suppose the case when Client 1 starts playing a news program in a train on 8:00 p.m. as shown in the left side of Fig. 2. The duration of the video is 5 minutes. One minutes after this, Client 2 starts playing the same video. In this case, the server delivers 2 video streams (a set of video data from the begging to the end) for 4 minutes from 8:01 to 8:05 in the simple conventional method. However, in our proposed technique, the server merges 2 streams from one minutes after the beginning to the end and delivers the merged stream to all clients as shown in the right side of Fig. 2. Client 2 can receive all data since it receives the data for one minutes from the beginning from the server directly. In this case, the server only delivers 2 streams for 1 minutes and can reduce the communication traffic. This is a simple example for the case of 2 clients. Actually, the server merges some streams for multiple clients. Broadcasting systems are suitable for delivering data to all clients. So, in our proposed technique, the server delivers the merged stream via the broadcasting system and other streams via the communication system. For this, we use hybrid broadcasting environments.

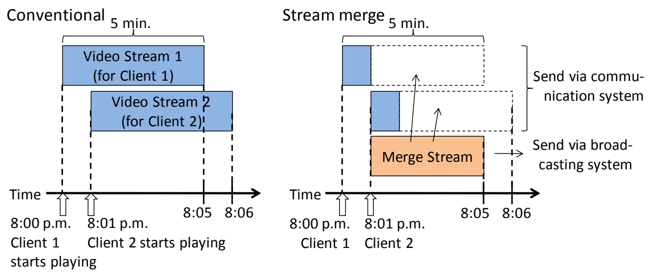

Figure 2: An example of stream merge

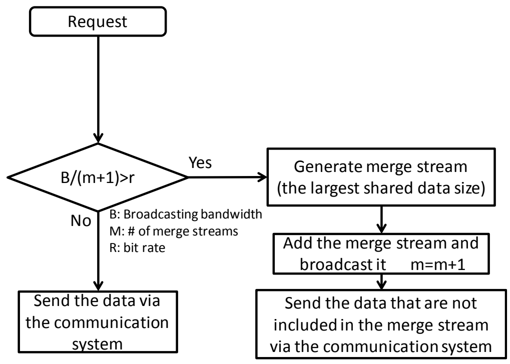

Figure 3: Flow chart for merging streams

Figure 3 shows the flow chart for merging streams in our proposed technique. B denotes the broadcasting bandwidth and m denotes the number of the currently merged streams. The broadcasting bandwidth for each merge stream in case of adding another merge stream is $B/(m+1)$. When a client requests playing the data, the server compares this value and the bit rate r. If this is larger than r, the server generates a merge stream and sends it via the broadcasting system since interruptions do not occur by receiving the merge stream. Otherwise, the server sends the requested data to the client via the communication system since interruptions can occur by adding a merge stream.

3.3 Spare Data Delivery

To solve the second problem, we propose the spare data delivery technique. In the spare data delivery, for the case when the condition of the electric wave gets worse, the system delivers spare data beforehand. Different from the traditional technique that the clients buffer the requested video data, spare data are the data to keep the motivation for the users to watch the video. We use two types of spare data.

One is a data not related to the requested video such as commercial or weather forecasting. For example, the mobile device plays a commercial movie when the user riding train enters a tunnel and cannot catch the electric wave. The clients can download such spare data before requesting playing videos since the contents of the spare data is not related to video requests. The clients play spare data if the video playback interrupts.

The other is data related to the requested video but the data size is small compared with the video played when the clients have enough bandwidth, e.g., text or static image data. For example, the mobile device shows news by text when the user

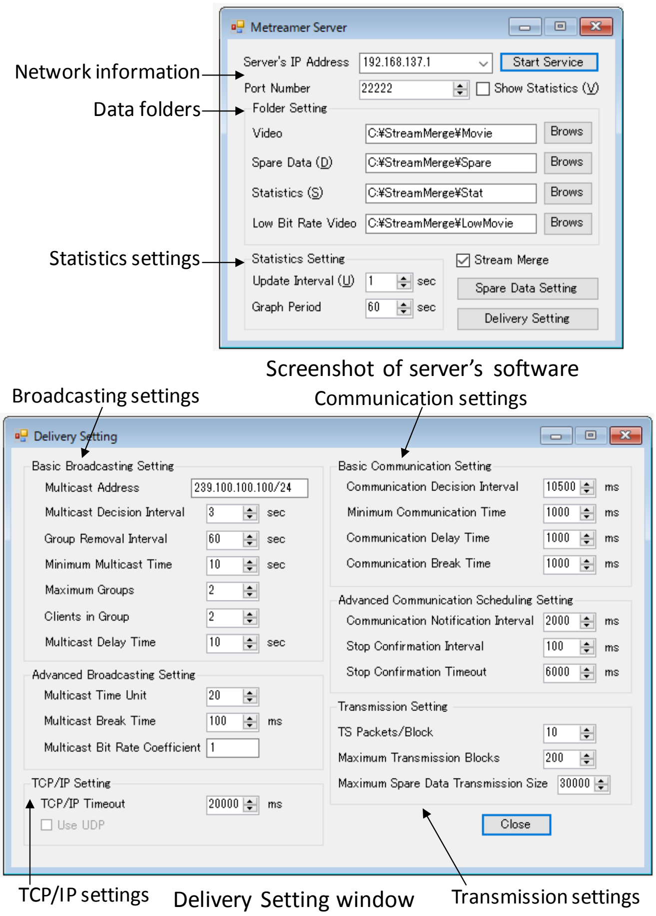

Figure 4: A screenshot of Metreamer server software

Figure 5: A mobile device running Metreamer

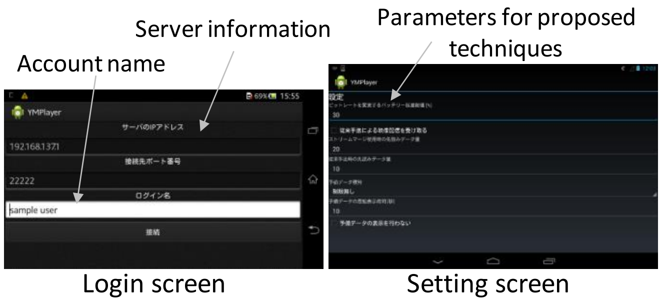

Figure 6: A screenshot of Metreamer client software

watches a news program and the video playback interrupts. The clients can download such spare data within a short time after requesting playing videos since the data size is very small.

When the users want to watch the whole requested video, the former type is suitable since the clients play the requested video although other videos can be played midstream. When the users want to grasp the content of the requested video, the latter type is suitable. In case of delivering the spare data not related to requests, broadcasting systems is suitable to deliver the data such as commercial or weather forecasting since these data are used as spare data for all clients. So, in our proposed technique, the server delivers such spare data via the broadcasting system on hybrid broadcasting environments. The detail algorithm such as how to get spare data and determine them depends on the implementation. We will explain the implementation for our developed system in the next section.

3.4 Remaining Battery based Bit Rate

To solve the last problem, we propose the remaining battery based bit rate. In the remaining battery based bit rate, the bit rate of the video is controlled on the client side. The clients play the video with normal bit rate when they have sufficient remaining battery to play the video. Otherwise, the clients reduce the battery consumption by decreasing the bit rate of the video. To change the bit rate on the client side, our proposed technique uses multi-bit rate encoding. In multi-bit rate encoding, the clients can play the video with some bit rates determined beforehand. The resolution or the size decreases when playing the video at a lower bit rate. For example, by using multi-bit rate encoding, the clients can play the video at 128Kbps even when the server delivers the video at 1Mbps bit rate. In our proposed technique, the users set their preferable remaining battery and the corresponding bit rate. The server delivers the video at the highest bit rate for all the bit rate used by the clients so as to decrease the traffic.

4 DEVELOPED SYSTEM

We develop a mobile video-on-demand system using our proposed techniques. We call our developed system Metreamer, the abbreviation of mobile ever streamer.

4.1 Overview

Metreamer uses the broadcasting address of UDP protocol, that sends the same data to all the clients connected to the same network, as the broadcasting system and the unicasting of TCP protocol as the communication system. The delivery server has the video data and the spare data. If the server sends the spare data at the same time with the video data when the clients request playing the video, interruptions easily occur. So, in Metreamer, the server sends the spare data when there is a remaining bandwidth capacity such as after stopping the service or finishing all video data deliveries. Metreamer can use video, image, and text data as the spare data. The video encoding type is widely used MPEG2. Metreamer can measure some statistic information such as the number of clients, and so on.

4.2 Software for Servers

Figure 4 shows a screenshot of the software for the servers of Metreamer. The software runs on Windows 7 and uppers. The server sometimes has some IP addresses and so the users can change the IP address on the software. The users can also change the directories for the video, spare, and statistic data. To show the statistic information, the users click the show button and set the interval to get the information. The users can set many other parameters for the video delivery as shown in the figure. By clicking the service start button, the software starts the delivery service and waits for the requests from the clients.

When the server software receives a request to play the video from a client, it start delivering the video data using our proposed stream merge technique explained in Subsection 3.2. Based on Fig. 3, the software sends the video data using the broadcasting address by UDP or the client's address by TCP. The software checks the connections from the clients cyclically and if a client disconnects for the reason of communication error or others, stops the video delivery. When the software finishes a video delivery to a client, it stores the statistic information. The software can show the information by graphs. When the software receives the request of receiving the spare data, it sends the data to the client.

4.3 Software for Clients

Figure 5 shows a mobile device playing a video using Metreamer. The mobile device is a special device for our p developed system. But the software for clients can run on other Android (4.1 or upper) mobile devices. The software for the clients of Metreamer first shows the login screen as shown in the left side of Fig. 6. By logging in, the client software can retrieve their settings. To enabling logging in, the users first register themselves to the server and after that they get IDs and passwords. The screen for setting some parameters is shown in the right side of Fig. 6. On this screen, the users can set the IP address of the server. In our developed system, the users directly input the server's IP address, but it can be set automatically by getting it from the server list wrote in some homepages in the Internet. On the setting screen, the users can set the data amount for cashing, the spare data types for the spare data delivery technique, the bit rates and the remaining battery to change the bit rate for remaining battery based bit rate.

After connecting to the server, the client software gets the video list and show the list. The users select their preferable video from the list and the software requests the video data by tapping the selected video. The software receives the data from the serer after the request, and the video playback starts when the software is ready for showing the video. If the software cannot receive the data for playing the subsequent video, it plays the spare data. In conventional systems, interruptions occur in such cases. The users can pause the video by tapping the pause button and stop the playback by tapping the stop button. When the users stop the playback or finish playing the video, the software again shows the video list.

4.4 Implementation of Proposed Techniques

4.4.1 Stream Merge Implementation

The server gets the broadcasting bandwidth B from its specification beforehand. If the actual broadcasting bandwidth largely differs from the specification, the server modifies the value. The server knows the number of merge stream N since it sends the merge streams. By checking the duration and the data size of each video, the server can get the bit rate r of the video data. Using these values, the server calculates $B/(m+1)$ when it receives the requests to play video from the clients. If the value is smaller than r, the server generates the video stream for the client and send it to the client. If the value is larger than r, the server generates a merge stream so that the duration of the merge stream becomes the longest. The merge streams are not merged to other merge stream again to make the merging algorithm simple in our developed system. Then, the server sends the merge stream to all clients via the broadcasting system and stops sending the streams that are merged to the merge stream. At the same time, the server sends the remaining data for the client, i.e., the data that the requested client cannot receive from the merge stream, to the client.

4.4.2 Spare Data Delivery Implementation

When a user runs the client software first, the client does not have spare data. So, the client requests the spare data to the server if it has no spare data. If the user changes the spare data type by setting screen, the client requests the spare data. The spare data are stored in the server and the clients receive some spare data from the server. Thus, the clients have some spare data before starting playing the video. If a client does not have the subsequent video and an interruption will occur, it selects a spare data randomly and shows it. If the number of the spare data is largely less than the number of the interruptions, the clients show the same spare data sometimes.

4.4.3 Remaining Battery based Bit Rate Implementation

The client software can get the remaining battery by asking it to the OS. When the remaining battery changes and the bit rate of the video changes according to the user's setting, the client changes the bit rate. If the client is playing a video, immediately changes the bit rate. Otherwise, the client uses the changed bit rate from the next video playback.

5 EXPERIMENTS

To investigate the effectiveness of our proposed techniques, we used our developed system in two practical situations since computer simulations do not reflect actual situations completely. We used our developed mobile video-on-demand system at Nakano central park and Okayama castle. Through these experiments, we got some questionnaire results. We report the experiments in this section.

Figure 7: Our experiment on Nakano central park

Figure 8: Our experiment on Okayama castle

5.1 Experiment at Nakano Central Park

With the cooperation of an industrial promotion organization in Nakano, we got a chance to provide a mobile video-on-demand service on practical field.

Before the full experiment, we did a preliminary experiment on July 2nd, 2014 at Nakano central park. In the preliminary experiment, we provided a mobile video-on-demand service using our developed Metreamer for 2 clients. We used the broadcasting equipment installed in the park. The broadcasting equipment uses wireless LAN. In the environment for the preliminary experiment, the client could receive the data both from the broadcasting system and the communication system when they were in the broadcasting area. In case where the clients were out of the broadcasting area, the clients could receive the data only from the communication system. We checked the number of interruptions using the measuring function of Metreamer and confirmed that Metreamer realized zero interruption in the case where the clients were in the broadcasting area. However, otherwise, interruptions frequently occurred. This was because many electric waves were emitted around the park and the communication bandwidth decreased down to the bit rate frequently. So, we decreased the bit rate of the video and did an experiment again on July 17th. In the experiment, Metreamer realized zero interruption even where the clients were out of the broadcasting area. To further investigate the performances of Metreamer, we did a full experiment on a large event.

Considering time and scale, the full experiment was done on Tohoku-Fukkou-Daisaiten-Nakano held on Oct. 25th and 26th, 2014 at the same place. The situations are shown in Fig. 7. For the experiment, we made the press release shown in the figure. To show the press release as it is, this is Japanese. The event is the largest one in Nakano area and the attendees are 170,000. Nebuta (large paper made statures) moves around the park during the event period. To deliver the video that is interesting for the attendees, we used the movie for moving Nebuta on the last year on the first day and that on the first day on the second day. The duration of the video was 2 minutes and 1 second, and the bit rate is 2Mbps. We set wireless LAN access point. The mobile devices of the attendees can receive the data both from the broadcasting system and the communication system by making them connect to the access point. We got booth, and there, the attendees watched the video using Metreamer. We asked the attendees answer the questionnaire after watching the video. We describe the detail of the questionnaire in Subsection 5.4. For the performance comparison, we also provide a video-on-demand service using conventional Windows Media System. In the environment for the full experiment, the video soon interrupted when the mobile devices started playing the video using the system. On the other hand, our developed Metreamer realized zero interruption even when about 10 clients connect to the system.

Table 1: System specification of Okayama experiment

Delivery Server

OS	Microsoft Windows 7 Professional SP1
CPU	Intel Core2 duo 2.5 GHz
Memory	4 GB
Ethernet	1000BASE-T

Wireless LAN

Access Point	Buffalo WAPM-AG300N
Wireless Standard	IEEE 802.11g
Additional Antenna	Buffalo WLE-HG-SEC

Clients for Demonstration

Google Nexsus 7 (2013, Android 4.3)
Sony Xperia C (C2305, Android 4.2.2)

Video Data

Coding	MPEG2-TS, H.264, AAC
Bit Rate	1Mbps
Frame Size	1280x720

5.2 Experiment at Okayama Castle

With the cooperation of Okayama city, we got a chance to provide a mobile video-on-demand service at Okayama castle. Considering time and scale, the experiment was done on Imagineering OKAYAMA ART PROJECT held on Nov. 22th, 2014 at Torishiro park in Okaya castle. This is the event to demonstrate arts made out and in Okayama. The situations are shown in Fig. 8. We set a wireless LAN access point and

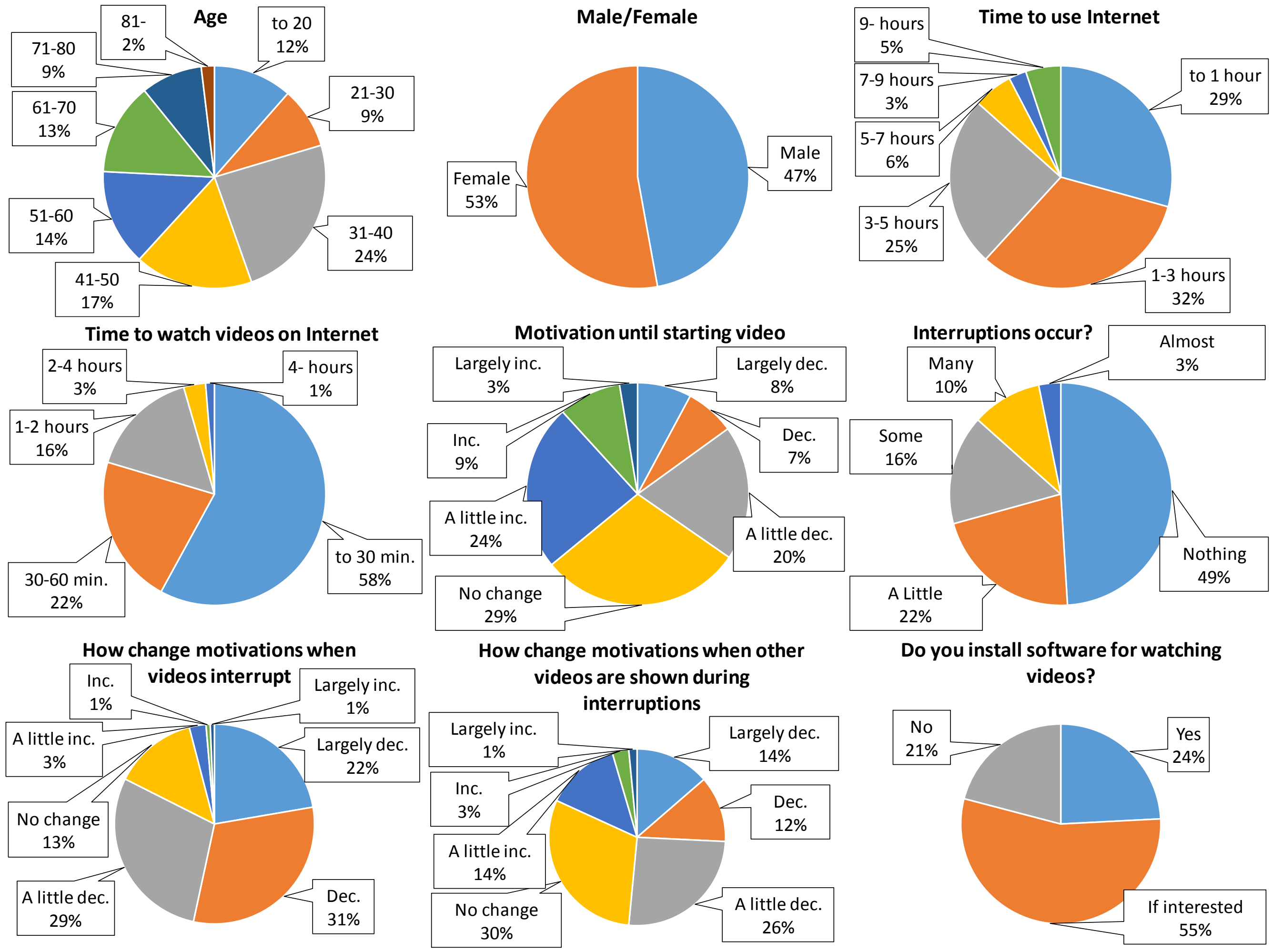

Figure 9: Questionnaire Results

booth in Okayama castle. Since Okayama castle does not have broadcasting equipment, we used the access point for the communication system and the broadcasting system. Table 1 shows the system specification for the experiment in Okayama. Okayama city has a PR of the city and uses it in some events. So, we delivered the video for the experiment. The duration was 3 minutes and 30 seconds and the bit rate is 1 Mbps. Same as the experiment at Nakano central park, the mobile devices of the attendees can receive the data both from the broadcasting system and the communication system by making them connect to the access point. We asked the attendees answer the questionnaire after watching the video at our booth. In this environment, the video soon interrupted when the mobile devices could not receive the data from the broadcasting system. On the other hand, our developed Metreamer realized zero interruption even when a few clients connect to the system.

5.3 Evaluation from Experiments

The experiments at Nakano central park was an experiments for a big event. Many people attended to the event during the period. Moreover, we delivered the video of moving Nebuta, which is the main event and many attendees were interested in. So, 16 clients connect to Metreamer at maximum. In the experiment, we made the situation that the clients could receive the data only from the communication system by moving them to the out of the broadcasting area. Even in this case, we confirmed that Metreamer realized zero interruption. Also we made the situation that the clients could receive the data only from the broadcasting system by disabling data transfer via the communication system. Even in this case, we confirmed that Metreamer realized zero interruption.

The experiments at Okayama castle was an experiments for a middle scale event. Only a few people were there sometimes. So, 7 clients connect to Metreamer at maximum. This is smaller than that of Nakano central park. Also in the experiment, we confirmed that Metreamer realized zero interruption even where the clients could receive the data only from the communication system or the broadcasting system.

5.4 Questionnaire Results

We got 275 questionnaire results from the experiments. The results are shown in Fig. 9.

First, regarding the statistics for the respondents, the most of the respondents' age is 31-40 years old and this is 24%. Female is more than male and is 53% of the respondents. The most of the time to use the Internet is 1-3 hours and is 32%.

In the respondents of using the Internet, 58% watch videos on the Internet up to 30 min and this is the most case.

To check the effectiveness of reducing the interruptions by the stream merge technology, we investigated the change of the motivation to watch videos. It takes a few seconds to start playing the video after the users tap the play button for the reason of starting the communication between the clients and the server. So, we asked the change of the motivation until staring playing the video. For the questionnaire, 29% of the respondents answered `no change'. We can say that the waiting time for starting playing the video after selecting the video does not influence the motivation largely. In the experiments, the video was not interrupted for 49% respondents. But, some respondents encountered the interruptions. In them, 82% of the respondents answered that their motivations decrease (largely, middle, a little) when the video interrupts. This is about 16 times more than that of increasing (total 5%). That is, our goal, reducing the interruptions, is significant to keep their motivations. One of the reasons why the motivation increases by the interruptions is that the users have an interest on the subsequent video contents. The motivations of 13% respondents did not change.

To check the effectiveness of the spare data delivery technique, we asked the change of the motivation when other videos are shown during interruptions. For the questionnaire, 52% of the respondents answered that their motivations decrease. The result is not a comparison with the video delivery without spare data delivery and the reason is simple. Their motivations decreased because of the video interrupts. If the system does not show the spare data during the interruptions, the ratio may increase. The motivations for 30% respondents did not change even when other videos are shown during interruptions. So, showing the spare data does not decrease the motivation largely.

It is difficult to check the effectiveness of the remaining battery based bit rate technique by questionnaires because the technique only changes the bit rate and the video quality decreases only thinking from the user side. However, we confirmed that the bit rate changes based on the remaining battery according to the user's setting and the technology worked on Metreamer.

To use Metreamer, the users have to install the software to receive the data from the server to their mobile devices. For the questionnaire, 55% of the respondents answered that they install the software if the system provides their interested videos. So, we can say that whether the users install a new software or not depends on the provided video contents.

6 CONCLUSION

Due to the recent proliferation of mobile devices and video-on-demand delivery, mobile video-on-demand delivery gets great attention. In this paper, aiming to zero interruption, we proposed 3 techniques for hybrid broadcasting environments. We developed a mobile video-on-demand system using our proposed techniques called Metreamer. In this paper, we reported the experiments of mobile video-on-demand delivery using our developed system.

In the future, we will again show the effectiveness of our proposed techniques by computer simulation. Also, we will develop the system for multiple streaming servers and live broadcasting.

REFERENCES

[1] V. Gopalakrishnan, B. Bhattacharjee, K. Ramakrishnan, R. Jana, and D. Srivastava, "CPM: Adaptive Video-on-Demand with Cooperative Peer Assists and Multicast," IEEE INFOCOM 2009, pp. 91-99 (2009).
[2] N. Magharei and R. Rejaie, "PRIME: Peer-to-Peer Receiver-Driven Mesh-based Streaming," Proc. of IEEE INFOCOM 2007, pp. 1415-1423 (2007).
[3] H. Kim and H. Y. Yeom, "Dynamic Scheme Transition Adaptable to Variable Video Popularity in a Digital Broadcast Network," IEEE Trans. on Multimedia, Vol. 11, No. 3, pp. 486-493 (2009).
[4] S. Kulkarni, J.-F. Paris, and P. Shah, "A Stream Tapping Protocol Involving Clients in the Distribution of Videos on Demand," Springer Advances in Multimedia, Special Issue on Collaboration and Optimization for Multimedia Communications, Vol. 2008 (2008).
[5] T. Taleb, N. Kato, and Y. Nemoto, "Neighbors-Buffering-Based Video-on-Demand Architecture," Signal Processing: Image Communication, Vol. 18, Issue 7, pp. 515-526 (2003).
[6] J. B. Kwon, "Proxy-Assisted Scalable Periodic Broadcasting of Videos for Heterogeneous Clients," Multimedia Tools and Applications, Springer, Vol. 51, No. 3, pp. 1105-1125 (2011).
[7] T. Yoshihisa, "A Data Segments Scheduling Method for Streaming Delivery on Hybrid Broadcasting Environments," Proc. of International Workshop on Informatics (IWIN2015), pp. 3-8 (2015).
[8] YouTube, http://www.youtube.com/.
[9] Ustream.tv:You're On, http://www.ustream.tv/.
[10] niconico, http://www.nicovideo.jp/.
[11] Netflix, http://www.netflix.com/.
[12] Hulu - Watch TV, Original, and Hit Movies, http://www.hulu.com/.
[13] Actvila, http://actvila.jp/.

(Received October 10, 2016)
(Revised April 8, 2017)

Tomoki Yoshihisa received his Bachelor's, Master's, and Doctor's degrees from Osaka University, Osaka, Japan, in 2002, 2003, 2005, respectively. Since 2005 to 2007, he was an assistant professor at Kyoto University. In January 2008, he joined Cybermedia Center, Osaka University as a senior lecturer and in March 2009, he became an associate professor. From April 2008 to August 2008, he was a visiting researcher at University of California, Irvine. His research interests include video-on-demand, broadcasting systems, and webcasts.

Tomoya Kawakami received his B.E. degree from Kinki University in 2005 and his M.I. and Ph.D. degrees from Osaka University in 2007 and 2013, respectively. From 2007 to March 2013 and from July 2014 to March 2015, he was a specially appointed researcher at Osaka University. From April 2013 to June 2014, he was a Ph.D. researcher at Kobe University. Since April 2015, he has been an assistant professor at Nara Institute of Science and Technology. His research interests include distributed systems, rule-based systems, and stream data processing. He is a member of the IPSJ and IEEE.

Yusuke Gotoh received a Bachelor's degree from Okayama University, Okayama, Japan, in 2005 and Master's and Doctor's degrees from Kyoto University, Kyoto, Japan, in 2007 and 2009, respectively. Since April 2009, he has been an Assistant Professor at the Department of Information Technology, Faculty of Engineering, Okayama University, Okayama, Japan. From April 2011 to November 2012, he was a Visiting Researcher at La Trobe University, Melbourne, Australia, as a JSPS Postdoctoral Fellow for Research Abroad. His research interests include broadcast computing, scheduling, and spatial computing.