Regular Paper

A Frame Rates Stabilization Mechanism for Trust-oriented Internet Live Video Distribution Systems

Tomoki Yoshihisa[*], Satoru Matsumoto[*], Tomoya Kawakami[**], and Yuuichi Teranishi[***,*]

[*]Cybermedia Center, Osaka University, Japan
[**]Nara Institute of Science and Technology, Japan
[***]National Institute of Information and Communications Technology, Japan
yoshihisa@cmc.osaka-u.ac.jp

Abstract - Due to the recent popularization of Internet live video distributions, Internet live video distributors such as YouTubers have attracted great attention. Some of them hide the image regions related to personal information, e.g., their faces or current locations, so as not to encounter public concerns such as threats or attacks to them. In the cases that the video processing time to hide these image regions is long, the frame rate fluctuates and unstable frame rates annoy the viewers. Hence, in this paper, we propose a frame rates stabilization mechanism for trust-oriented Internet live video distribution Systems[1]. Our proposed mechanism adopts two approaches. One is changing the image region for video processes. The other is changing the video process when the video processing time is going to exceed the interval of the video frame. Our evaluation results revealed that our developed system with the proposed mechanism can stabilize the frame rate for trust-oriented Internet live video distributions.
Keywords: Broadcasting, YouTubers, Continuous Media, Data Streaming, Video-on-Demand

1 INTRODUCTION

Due to the recent popularization of Internet live video distributions, Internet live video distributors have attracted great attention. Most of Internet live distributors distribute the videos shooting themselves by cameras. For example, video distributors on YouTube are called YouTuber and 1 million of YouTubers distribute the videos shooting themselves. They often distribute live videos. Some of them hide image regions related to personal information, e.g., their faces or current locations, so as not to encounter public concerns such as threats or attacks by the viewers. It has a large possibility that public concerns do not occur if there is a trust between Internet live video distributors and the viewers because the trust construct their social relations. Therefore, trust-oriented Internet live video distributions proposed in [1] have a large possibility to realize safer and wider used Internet live video distributions.

In the cases that the distributors shoot themselves, the most sensitive personal information is their faces. To avoid public concerns, some of them change or hide their faces by adding video effects [2]. Such video effects include some processes for detecting their faces, creating mask images, and drawing the mask images to their faces, and have a higher computational load compared with a simple process. In the cases that the video processing time is longer than the interval of the video frames, the time to draw the processed image for a video frame delays and the frame rate decreases. The video processing times depend on the complexity of the processes and the images, and the delays cause unstable frame rates. Unstable frame rates annoy the viewers. Therefore, stable frame rates are required for Internet live video distributions.

Various techniques to reduce video processing time have been proposed. Some of them give an upper limit on the video processing time and cancel the process when the processing time reaches to the upper limit. In the cases that the video process is changing or hiding the distributor's face, the face appears in the video when the processing time reaches to the upper limit because the process is cancelled. This is not a trust-oriented Internet live video distribution since the video exposes the distributor's personal information even when there is no trust between the distributor and the viewers. In trust-oriented Internet live video distributions, distributors' personal information should be hidden when there is no trust. However, existing video processing time reduction techniques do not consider trust and cannot realize trust-oriented Internet live video distributions.

In this paper, we propose a frame rates stabilization mechanism for trust-oriented Internet live video distribution systems. In trust-oriented Internet live video distributions, the distribution situation is classified into two situations. One is the situation that there is no trust between the distributor and the viewers (un-trusty). The other one is the situation that there is trust (trusty). In Figure 1, the left image shows the left image shows the case of an un-trusty situation and the distributor's face is blurred to hide his personal information. The right image shows the case of a trusty situation and no areas are blurred. In the un-trusty situations, our proposed system always hide the distributor's personal information. To achieve this, our proposed mechanism changes the video processes when the video processing time is going to exceed the interval of the video frame to a simple process. For example, in the cases that the time is close to draw the image for the next frame while the processing computer executes the process to detect the distributor's face in the video, the computer cancels the face detection process and starts executing the

[1] The work was supported by a Grants-in-Aid for Scientific Research (C) numbered JP18K11316, and by I-O DATA Found.

Figure 1: Live video images for an un-trusty situation (left) and a trusty situation (right).

process to blur the whole region of the image. In addition, we develop a system with our proposed mechanism and evaluate its performance. The main novelty of the paper is the stabilization of the frame rates considering trust on Internet live video distribution. Although some previous researches try to stabilize frame rates, they do not consider trust and wholly give up to execute video processing when the processing time reaches to the upper limit. Our proposed system change the process considering to manage the trust. The contributions of this research are; 1) the proposition of a frame rates stabilization mechanism for trust-oriented Internet live video distribution systems, 2) the design and the development of a system with the mechanism, 3) the evaluation of the system.

The result of the paper is organized as follows. We introduce related work in Section 2. We explain the trust-oriented Internet live video distributions in Section 3 and our proposed video processing system to stabilize frame rates in Section 4. We show some evaluation results and discuss about them in Section 5. Finally, we will conclude the paper in Section 6.

2 RELATED WORK

Many systems have been proposed for distributed stream processing (DSP) in a peer-to-peer (P2P), cloud, edge, or fog computing model [3]-[13]. Some of the proposed systems assume or are implemented by open-source stream processing platforms such as Apache Hadoop [14], Storm [15], Flink [16], and Spark Streaming [17]. Lopez et al. carried out two experiments concerning threats detection on network traffic to evaluate the throughput efficiency and the resilience to node failures for Storm, Flink, and Spark Streaming [18]. The results show that the performance of native stream processing systems, Storm and Flink, is up to 15 times higher than the micro-batch processing system, Spark Streaming. On the other hand, Spark Streaming was robust to node failures and provided recovery without losses.

Some of the DSP systems are designed for real-time processing and can be applied for live video streaming. In [5], the proposed scheme determines the evaluation order for the conditional expressions in continuous queries to reduce the processing time. Its evaluation shows that the proposed scheme can reduce the maximum number of communication hops and the average amount of communication traffic in IoT environments. Ning et al. proposed Mobile Storm as a distributed real-time stream processing system for mobile cloud [7]. Without offloading computation to remote servers, Mobile Storm processes real-time streaming data using a cluster of mobile devices in a local network. Mobile Storm was implemented on Android phones, and a video stream processing application was developed to evaluate its performance. The

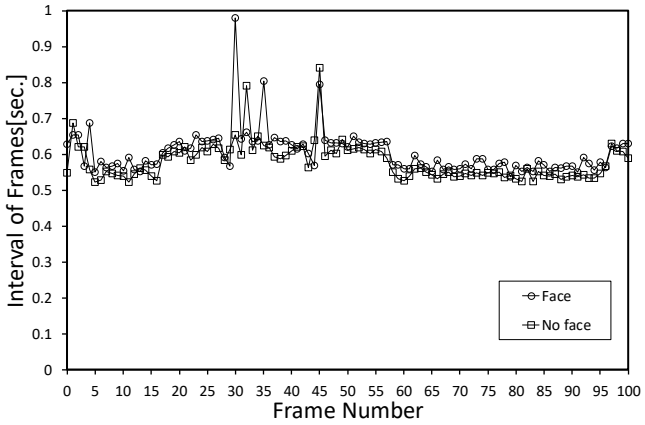


Figure 2: Example intervals of frames

results show that Mobile Storm is capable of handling video streams of various frame rates and resolutions in real-time. Choi et al. proposed DART as a fast and lightweight stream processing framework for the IoT [9]. In DART, a logical group of data sources, namely, a Cloud of Things (CoT) is composed to process the data streams more efficiently in a fully distributed fashion. DART aims to overcome both server-based and edge-only-based methods by grouping IoT devices as a CoT. RIDE was proposed to process real-time massive image stream on distributed environment efficiently [10]. RIDE consists of four layers: application, master, buffer, and worker layers. To minimize the communication overhead between the tasks on distributed nodes, coarse-grained parallelism is achieved by allocating partitions of streams to worker nodes in RIDE. In addition, fine-grained parallelism is achieved by parallel processing of task on each worker node. Yang et al. focused on distributed fault-tolerant processing (DFP) method and proposed a distributed image-retrieval method designed for cloud-computing based multi-camera system in smart city [11]. Through the combination of the cloud storage technology, data encryption, and data retrieval technology, efficient integration and management of multi-camera resources are achieved. In [12], processing and bandwidth issues for a typical video analytics application were investigated to help understand placement decision of methods between edge and cloud. The authors in [12] also say that there are further considerations than made in [12], such as privacy, central sharing, edge device maintenance, and processing and bandwidth costs.

Related to security or privacy of real-time video data, Liu et al. proposed an infrastructure for secure sharing and searching for real-time video data [19]. The proposed infrastructure is particularly suitable for mobile users by deploying 5G technology and a cloud computing platform. Its security is guaranteed even if the cloud server is hacked since data confidentiality is protected by cryptographic encryption algorithms. In addition, the proposed infrastructure also provides secure searching functionality within a user's own video data. Wang et al. proposed OpenFace, an open-source face recognizer whose accuracy approaches that of the best available proprietary recognizers [20]. Integrating OpenFace with interframe tracking, they also built RTFace, a mechanism for denaturing video streams that selectively blurs faces according to speci-

fied policies at full frame rates. In [20], privacy-aware architecture for large camera networks using RTFace were presented.

The existing techniques mentioned above can reduce the stream processing time by efficiently distributing the computational and communication loads to the processing nodes. However, those techniques do not give an upper limit for the processing time and there is a large possibility that the frame rate fluctuates. This paper is positioned to the consideration of the situations such as allowed processing time and adaptive task modification on the trust-oriented Internet live video distribution.

3 TRUST-ORIENTED INTERNET LIVE VIDEO DISTRIBUTIONS

In this section, we briefly explain trust-oriented Internet live video distributions proposed in [1].

In trust-oriented Internet live video distributions, the distribution situation is classified into two situations. One is the situation that there is no trust between the distributor and the viewers and is called the *un-trusty* situations. The other one is the situation that there is trust and is called the *trusty* situation. The situation that an Internet live video distribution belongs to depends on various factors. For example, in the cases that the viewers of an Internet live video distribution is limited to only the friends of the distributor, the situation is a trusty situation. In the cases that the viewers are the stranger of the distributor, the situation is an un-trusty situation. In a simple system, the un-trusty/trusty situation that the current situation belongs to is selected by the distributor manually. Automatic selection is also possible by using the information about the viewers, the locations, and so on. Figure 1 shows example images of live video in an un-trusty situation and a trusty situation.

The trust-oriented Internet live video distributions have three policies, “close-information”, “limit-information”, and “expose-information” policies. In the un-trusty situation, the distributors may use the close-information policy. In this policy, the processing computer executes the processes so as to close personal information. The distributor can safety distribute the shot live video since the distributor’s personal information is closed. However, the possibility to be able to construct trust decreases since the viewers cannot get the personal information about the distributor so any more. In the limit-information policy, the processing computer executes the processes accepted by the distributor. The distributors select this policy aiming to keep the trust. In the trust situation, the distributors may use the expose-information policy. In this policy, the processing computer executes the processes so as to expose personal information. The possibility to be able to construct trust increases since the viewers can get the personal information about the distributor. However, the live video distribution is unsafe.

In the trust-oriented Internet live video distribution, the processing computer executes video processes based on the policy selected by the distributor (un-trusty or trusty). However, conventional trust-oriented Internet live video distribution systems do not consider the processing time and the frame rate usually fluctuate.

4 PROPOSED MECHANISM

We explain our proposed frame rates stabilization mechanism in this section. We explain our target problem first and our approach to solve the problem after that.

4.1 Target Problem

As described in Section 1, long video processing times cause unstable frame rates and this annoy the viewers. For example, we show the intervals of the frames in Fig. 2. The horizontal axis is the frame number. In this example, the processing computer executes the processes to detect faces and to blur the detected region. “Face” indicates the interval of the frames when there is a distributor’s face in the live video. “No face” indicates that when there is no distributor’s face. As shown in this example, the interval of the frames fluctuate and have a range of approximately 200 [msec.] in this case. Such a large change of the interval has a large possibility to annoy the viewers. Therefore, in this paper, we propose a frame rates stabilization mechanism.

In the cases that the processing time is shorter than the interval of the frames, the system can control the time to draw the image for the next frame by waiting for some time. Otherwise, the time to draw the image for the next frame delays and the frame rate changes. We propose a frame rates stabilization mechanism even when the processing time is long.

4.2 Our Approach

To stabilize frame rates for trust-oriented Internet live video distributions, we adopt two approaches.

The first one is the reduction of the video processing time. As explained in the previous subsection, the system can stabilize the frame rate if the processing time is shorter than the interval of the frames. Therefore, shorter video processing times than the interval of the frames enables stable frame rates. Various techniques to reduce video processing time has been proposed. However, they do not focus on the trust-oriented Internet live distributions and some public concerns can occur even when the system adopts these techniques. The most sensitive personal information is their faces. Therefore, we focus on hiding the distributor’s faces and propose a video processing time reduction method for hiding the face in the trust-oriented Internet live video distributions.

The other one is the change of the video process to a simple video process. As explained in Section 1, most of video effects require a higher computational load compared with a simple process. Therefore, our proposed method changes the video process when the video processing time is going to exceed the interval of the video frames to a video effect that the processing computer can execute the process with a shorter processing time. An example of a simple video process is blurring the whole region of the image. Since the processing time of a simple video process is relatively short compared with complicated video processes, the system can finish the process before the time to draw the image for the next frame and thus can stabilize the frame rate.

We will explain the detail of each approach in Subsections 4.4 and 4.5.

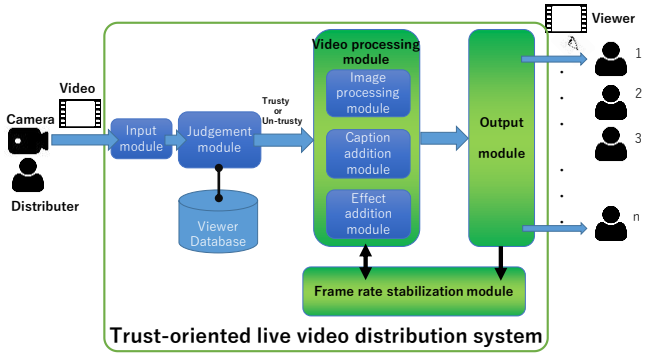


Figure 3: Our assumed system for our proposed mechanism

4.3 Assumed System

Figure 3 shows our assumed system for our proposed mechanism. In the system, the distributors distribute their shot live videos to the viewers using the trust-oriented live video distribution system. The situation judgement module judges whether the current situation is an un-trusty or a trusty situation using the viewer database and other information. The video processing module executes the designated video processes. A part of the viewers login to the system before watching the live videos. The details of these modules are inscribed in [1].

The frame rate stabilization module is newly added to the system. The frame rate stabilization module manages the video processing module and checks the frame rates of the output module. The output module is the system module to distribute the live videos to the viewers. When the frame rate is going to fluctuate, the frame rate stabilization module changes the video process executed in the video processing module to a simple process and try to reduce the processing time. The processed video data are transferred to the output module and are distributed to the viewers.

4.4 A Method to Hide Faces Faster

In this subsection, we explain our proposed method to hide the distributor's face faster. Our proposed method does not fix the face detection algorithm. Various algorithms that were already proposed can be adopted to our proposed method.

4.4.1 Position of Face

Most of live video distributors shoot themselves and the position of the face does not change largely. We do not assume that the position of the face does not change. For example, the distributor fixes the camera to his room by a tripod and shoots himself sitting on the front of the camera. In this case, the position of his face in the video image does not change largely although the position waves when he moves his upper body. For another example, the distributor uses her smartphone to distribute the live video. She grasps her smartphone forwarding the camera to herself and shoots herself while walking. The distributors generally allocate their faces to the center of the video image and the position does not change largely and frequently.

In the cases that the position of the face does not change largely, the position of the face in the next frame image is close to that in the current frame image. By shrinking the range to detect the faces using this feature, we can reduce the video processing time since a smaller image gives a shorter processing time. However, a smaller range to detect the faces has a large possibility to fail to detect the faces since the possibility that the shrunk image does not include the face increases. Therefore, our proposed method takes a margin from the position of the face in the current frame image and detects the face in the range.

The appropriate margin length is the length that the position of the face changes within the interval of frames. If the position changes to the out of the detection area even taking the margins, the face detection fails.

4.4.2 Region to Detect Face

Let X_c, Y_c denote the upper left corner of the region for the detected face in the current video frame image and W_c, H_c denote the width and the height of the region. We set the same margin to the right and the left sides of the region (M_X) and to the top and the bottom sides of the region (M_Y). Then, the region to detect the faces in the next video frame image X_d, Y_d, W_d, H_d are given by:

$$\begin{aligned} X_d &= X_c - M_X \\ Y_d &= Y_c - M_Y \\ W_d &= W_c + 2M_X \\ H_d &= H_c + 2M_Y \end{aligned} \quad (1)$$

In our proposed method, the region to detect the face is determined by the above equations. In the evaluation section, we will confirm the effectiveness of shrinking the region to detect the face.

4.5 A Method to Change Video Process

In this subsection, we will explain our proposed method to change video process to stabilize the frame rates.

4.5.1 Timing to Change Video Process

In our proposed mechanism, the frame rate stabilization module changes the video process when the processing time is going to exceed the interval of the frame to a simple process that the video processing module can execute the process with a shorter processing time. Even when the video process is changed to the simple one, the video processing time arises and it takes some time. Therefore, our proposed method gives a margin time for the simple video process. Let T_m denote the margin time. When the current time satisfies the following inequality, the frame rate stabilization module changes the video process to the simple one.

$$T_c > T_{nf} - T_m \quad (2)$$

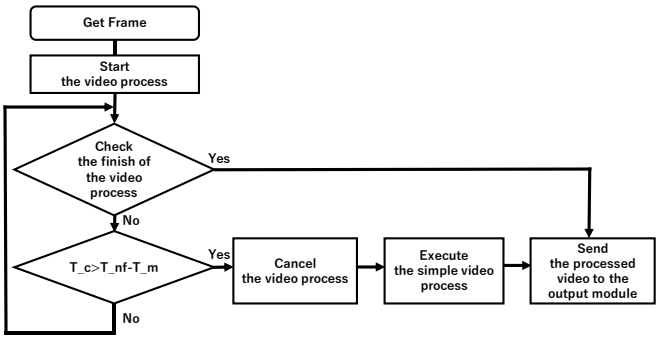


Figure 4: The flowchart for the frame rate stabilization module

Here, T_c is the current time and T_{nf} is the time to draw the next frame image. That is, the value of $T_{nf} - T_m$ is the *deadline* of the original video process. Our proposed method decides T_m enough to finish the simple video process within the margin time. For example, in our brief experiment, the maximum processing time to blur the whole region in a video processing system is 129 [msec.]. In this case, we can predict the sufficient time to finish the simple video process although the time fluctuates. In this case, the value of T_m larger than 129 [msec.] is sufficient.

In the cases that the right formula of Equation (2) is very close to the left, the video process can frequently change. This can cause the annoyance of the views. To determine T_m, it is better to consider such an influence of the viewers.

4.5.2 Video Process to Hide Face

Figure 4 shows the flowchart for the frame rate stabilization module. When the module gets a video frame image from the camera, it starts the video processing to the image in other thread. To manage the processes, the video processes are executed in parallel and in other thread from the frame rate stabilization module. After that, the module continuously checks whether the process finishes and whether the above inequality is satisfied or not. When the process finishes, the module transfers the processed image for the video frame to the output module. When the above inequality is satisfied, the module cancels the video process and start executing the simple video process. When the simple video process finishes, it transfers the processes image for the video frame to the output module.

5 EVALUATION

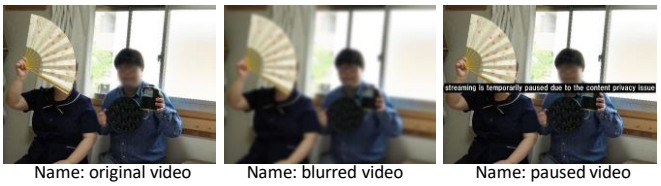
In this section, we investigate the influence of changing the video processes and show the results in Subsection 5.1. After that, to evaluate the performance of our developed system with the proposed mechanism, we show some evaluation results about the intervals of the frames and the video processing.

5.1 Influence of Changing Video Process

The objective of the evaluation is to check which video gives a higher QoE, i.e., the viewers can watch the video without a more annoyance, in un-trusty situations. For this objective, the videos shown to the subjects are not necessarily live videos. To show the same video to all subjects, therefore, we shoot a video and create three videos from it although our research target is live videos. In this section, to evaluate the performance of our developed system with the proposed mechanism, we show some evaluation results about the intervals of the frames and the video processing.

Table 1: The videos created for the subjective evaluation

Video name	Description
Original	Human faces are hidden.
Blurred	The whole region is blurred after 10 [sec.] from the beginning.
Paused	The video is paused after 10 [sec.] from the beginning.



Name: original video Name: blurred video Name: paused video

Figure 5: The videos for the subjective evaluation

Table 2: The subjective evaluation result

Video name	1st	2nd	3rd
Original	11	0	0
Blurred (proposed)	0	9	2
Paused	0	2	9

5.1.1 Subjective Evaluation Setting

A realistic scenario to which our proposed trust oriented live video distribution is applied is the situation that the distributor fixes the camera to his room by a tripod and shoots himself siting on the front of the camera as explained in Subsection 4.4.1. Moreover, to avoid the exposure of the distributor's personal information in un-trusty situations, we blur the human faces and hid them. We call this the *original* video. The duration is 30 [sec.] Table 1 shows the explanations of our created three videos. The *blurred* video is the video created from the original video to simulate our proposed mechanism. To investigate the influence of changing the video processes, we blur the whole region of the video after 10 [sec.] from the beginning to the end. Only the human face is blurred by 10 [sec.] from the beginning and thus the beginning part of the video is the same as the original video. The *paused* video is the video created from the original video to simulate pausing the video when the load for the video processes is high. During the pausing, the text message "streaming is temporarily paused due to the content privacy issue" is shown in the center of the video. In the video, we pause the video after 10 [sec.] from the beginning to the end. Similar to the blurred video, the beginning part of the video is the same as the original video. The audios for all the videos were not interrupted and kept playing. Figure 5 shows a screen shot of the videos.

We show these videos to 11 subjects and ask the subjects to rank the videos in the order that he/she can watch the video

without a more annoyance. Also, we get the reason for their ranking.

5.1.2 Subjective Evaluation Result

Table 2 shows the result. From the result, we can see that the rank of our proposed mechanism is higher than the paused video. The summary of the reason was that the blurred video kept playing although the whole region is blurred, and thus the subjects could somehow grasp the contents of the video compared with the cases of the paused video. However, two subjects gave a higher rank to the paused video than the blurred video. The summary of the reason was that the text message that explained the reason why the video was paused was shown in the paused video. Thus, the subjects could understand why the video was paused and they did not get annoyance further than the blurred video. We think that this can be improved by adding a text message that explains the reason why the whole region is blurred to the blurred video. All subjets gave the highest rank to the original video because the case of the original video did not encounter any troubles.

5.2 Performance Evaluation Setting

From this subsection, we evaluate the performance of our developed system. We developed a video processing system that detects the faces in the shot video images and blurs the detected region. In the cases that the time to finish the video process is longer than the deadline (Equation (2)), the system cancels the processing and changes it to the simple process that blurs the whole region of the image. The system has a function to distribute the live video, but the function is not directly related to this research and we do not focus on this function.

To show the effectiveness of our proposed method, we show the intervals of the frames. We set the frame rate of the video to f [fps] in this evaluation. In our proposed method, when the current time exceeds the deadline, the processing computer changes the video process to the simple one. Therefore, in the cases that the interval of the frame is shorter than the $1/f$ [msec.], our proposed method can achieve a constant frame rate by waiting for drawing the next frame image.

We use the ratio of changing process in this evaluation. The ratio means the ratio that the video process is changed to the simple one and is the number of the simple video process divided by the number of the all video process. A larger value means that a more number of the video frames is wholly blurred (the whole region is blurred). A smaller value is better for the trust-oriented Internet live distributions since just for the region related to the personal information is blurred.

5.3 Evaluation Environment

For the evaluation, we use our developed video processing system. We developed the system using the Visual Studio 2017 and the system uses OpenCV 4.1.0 to get the images from the camera and to detect the faces in the images. The processing computer is a laptop computer (CPU: Core i7@2.4GHz, Memory: 8GB). We use the camera equipped on the laptop and get 640x480 RGB (32bits) images. For the face detection, we use the detectMultiScale function implemented in the OpenCV.

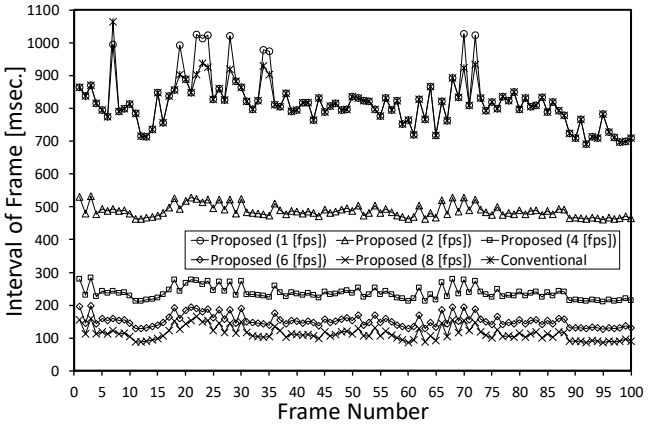


Figure 6: The frame number and the intervals of the frames

5.4 Evaluation Results

In this subsection, we show some evaluation results. First, we show the intervals of the frames and evaluate the effectiveness of our two approaches. After that, we show the performances changing the region sizes for the processes to evaluate our approach to change the region size to hide the face faster. Then, we show the performances changing some parameters for the face detection. We evaluate our developed system under the situations that the video records just one faces although the system can detect multiple faces to make the evaluation results easily understandable. In the situations that the video records multiple faces, their positions also influence the results.

5.3.1 Interval of Frame

The target problem of this research is the stabilization of the frame rate. This means that the intervals of the frames are more constant. Therefore, we measured the intervals of the frames.

Figure 6 shows the intervals of the frames. The horizontal axis is the frame number and the vertical axis is the intervals of the frames. In the figure, "Proposed (f [fps])" indicates the intervals of the frames under our proposed method when the frame rate is set to f. "Conventional" indicates the intervals of the frames under conventional methods, i.e., without our proposed method. The result under the conventional method does not depend on the frame rate since the method does not consider the frame rate. The margin time is 100 [msec.]

From this result, we can see that our proposed method achieves a shorter interval than the inverse value of the frame rate in many cases. This means that our proposed method gives stable frame rates. However, the intervals of the frames are sometimes longer in the cases that the video processing time is longer than the predicted margin time. The conventional method can achieve almost 1 [fps]. In the cases that the frame rate is 1 [fps], some intervals of the frames under our proposed method is the same as that under the conventional method because the video process finishes before the deadline. However, in the cases that the video process before changing it finishes earlier than the simple video process, the

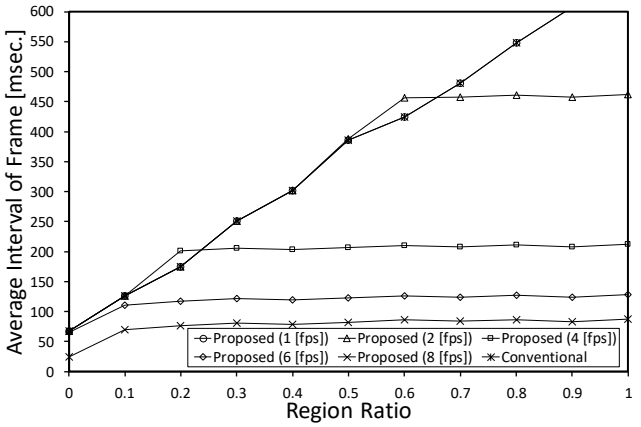


Figure 7: The average interval of the frames under the different region sizes for the processes

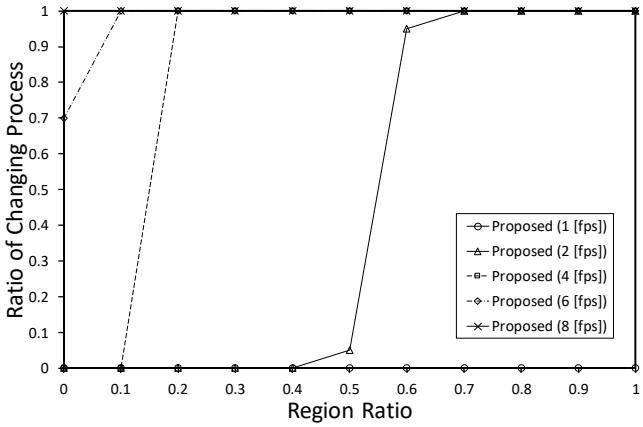


Figure 8: The ratio of changing the processes under the different region sizes for the processes

intervals of the frames under our proposed method is longer than that under the conventional method. For example, in the case of 1 [fps], the video processing times of 6 frames in 100 frames exceed the time to draw the image for the next frame. In the case of 4 [fps], the video processing times of 19 frames in 100 frames exceed the time to draw the image for the next frame. We can reduce the number of the frames in that the processing time exceeds the time to draw the image for the next frame by increasing the margin time.

The ratio of changing the process increases in proportional to the frame rate since a larger frame rate gives an earlier deadline. For example, in the cases of 1 [fps], 10 video processes are changed to the simple video processes in 100 frames and this means the ratio of changing the process is 10/100=0.1. In the cases of 4 [fps], all video processes are changed to the simple video processes and the ratio is 1.0. Therefore, Our proposed method gives stable frame rates but the ratio of changing process increases.

In the following evaluation results, we use the average interval of frame.

5.3.2 Region Size for Process

A smaller region size for the process gives a shorter video processing time because the data amount for executing the video process decreases. Hence, changing the region size for the process, we measured the average interval of the frames and the ratio of changing the processes.

Figure 7 shows the average intervals of the frames under the different region sizes for the processes. The horizontal axis is the region ratio. The region ratio is the ratio of the region for the process in the whole region and given by the number of the pixels in the region for the process divided by that in the whole region. The vertical axis is the average intervals of the frames. The margin time is 100 [msec.]. The legends are similar to the previous results.

In our proposed method, when the region ratio is small, the average interval of the frames increases as the region ratio increases since the data amount for executing the video process increases. When the region ratio is large, our proposed method gives almost constant average interval of the frames since the video process is changed to the simple video process when the video processing time reaches to the deadline. The video processing time earlier reaches to the deadline as the frame rate increases. The average interval of the frames under the conventional method increases in proportional to the region ratio, since the method does not consider the deadline. For example, the average interval of the frames in the cases of 4 [fps] is approximately 200 [msec.] This is smaller than 250 [msec.] (the interval of the frames under 4 [fps]) and our proposed method can give a stable frame rate. On the other hand, in the conventional method, the region ratio should be less than 0.23 to achieve the frame rate of 4 [fps].

Figure 8 shows the ratio of changing the processes under the different region sizes for the processes. The horizontal axis is the region ratio and the vertical axis is the average interval of the frames. In the conventional method, the processing computer does not change the video process and always execute the process to detect the faces. Therefore, the ratio of changing the process is always zero and we do not show it in the figure.

In the case of 2 [fps], the ratio of changing the processes increases sharply when the region ratio is 0.6. This is because the video processing time for detecting the face in the frame image is almost constant and most of all video processing time exceeds the deadline when the region ratio is larger than 0.5. A similar situation occurs in the case of 4 [fps] and the ratio of changing the processes sharply increases when the region ratio is 0.2. In the cases of 8 [fps] and larger frame rates, the ratio of changing processes are always 1.0 because the video processing time exceeds the time to draw the image for the next frame even if the video process changes to the simple video process.

5.3.3 Number of Neighbors

One of the main parameters for the face detection techniques is the number of the neighbors. General face detection techniques moves the target region to detect the face in the original image with changing the size of the target region and compares the image in each target region with a template image for faces. Therefore, one face in the original image is detected several times as shown in Fig. 9. To reduce the error for the face detections, the region in that some faces are detected within the close range is finally defined as the region of the face. This is called the neighbors and the number of the

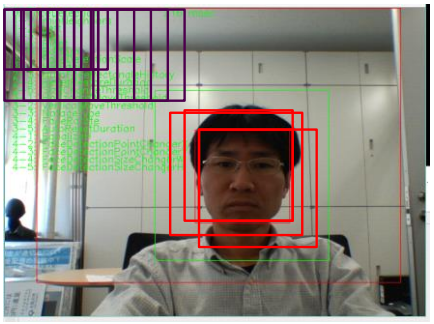


Figure 9: The image of detecting faces

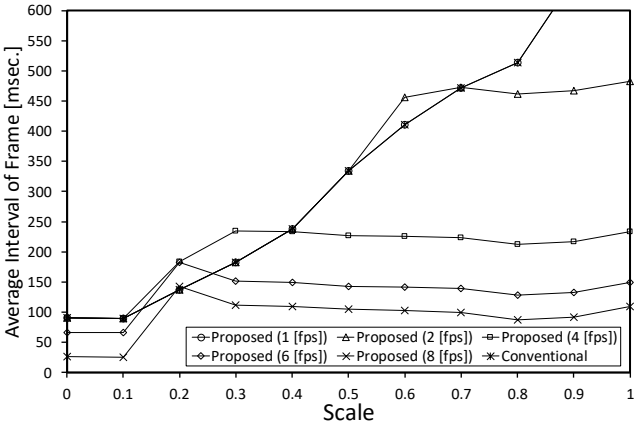


Figure 11: The average intervals of the frames under the different sizes of the images

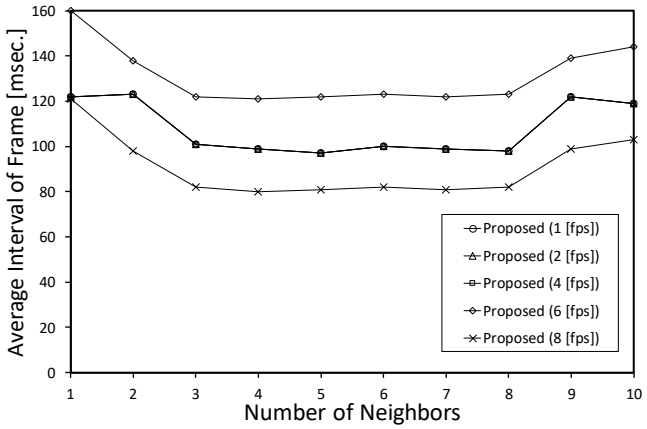


Figure 10: The average intervals of the frames under the different number of the neighbors

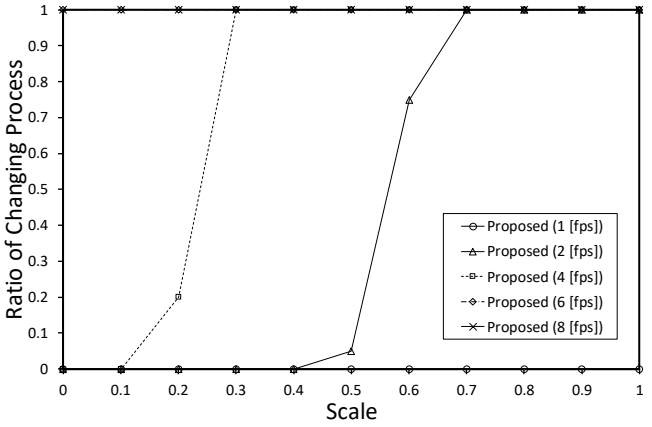


Figure 12: The ratios of changing the processes under the different sizes of the images

neighbors has a possibility to influence the video processing time. Hence, we calculated the average interval of the frames changing the number of the neighbors.

Figure 10 shows the average interval of the frames under the different number of the neighbors. The margin time is 100 [msec]. The region size for the video process is 320x240. From this figure, we can see that the average interval of the frames does not change largely even when the number of the neibhbors changes. This is because the algorithm we used detects all regions of the faces in the whole image and counts up the number of the neighbors after that. Thereore, the number of the neighbors has not so large influence on the average interval of the frames. The ratio of changing processes is 0.0 when the frame rate is 1, 2, 4 [fps] and is 1.0 when the frame rate is 6, 8 [fps]. The result is very simple and we do not show it here.

5.3.4 Scale of Image

As we explained in the previous section, a smaller size of the region for the process gives a shorter video processing time because the data amount for executing the video process decreases. In this section, changing the size of the image for the video process, we measured the average interval of the frames and the ratio of changing the processes.

Figure 11 shows the average intervals of the frames under the different sizes of the images. The horizontal axis is the scale. The scale is the ratio of the size of the image for the video process compared with the original image size. For example, when the scale is 0.5, the image downsizes to 320x240 in the cases that the original image size is 640x480. The vertical axis is the average intervals of the frames. The margin time is 100 [msec]. The region size for the process is 640x480. Similar to the results changing the region size for the process, when the scale is small, the average interval of the frames increases as the scale increases.

Figure 12 shows the ratio of changing the processes under the different sizes of the images. This is also similar to the result changing the region size for the process. The ratio sharply increases since almost video processing time exceeds the deadline under a large value of the scale.

6 CONCLUSION

In this paper, we proposed a frame rates stabilization mechanism for trust-oriented Internet live video distribution systems. Our proposed mechanism changes the image region for video process and also changes the video process when the video processing time is going to exceed the interval of the video frame. We developed a system with our proposed mechanism. Our evaluation results revealed that our developed system can stabilize the frame rate for trust-oriented Internet live video distributions. In the future, we will further

reduce the video processing time and investigate the frame rates in practical Internet live video distributions.

REFERENCES

[1] T. Yoshihisa, S. Matsumoto, T. Kawakami, and Y. Teranishi, "Trust-oriented Live Video Distribution Architecture", in Proc. IEEE Int'l Conf. on Computers, Software and Applications, to appear (2019).

[2] S. Matsumoto, T. Yoshihisa, T. Kawakami, and Y. Teranishi, "A Distributed Multi-Viewpoint Internet Live Broadcasting System with Video Effects", in Proc. International Workshop on Informatics (IWIN'18), pp. 83-88 (2018).

[3] X. Zhao, H. Ma, H. Zhang, Y. Tang, and Y. Kou, "HVPI: Extending Hadoop to Support Video Analytic Applications", in Proceedings of the 8th IEEE International Conference on Cloud Computing (CLOUD 2015), pp. 789-796 (2015).

[4] W. Kou, H. Li, and K. Zhou, "Turning Video Resource Management into Cloud Computing", Future Internet 8(3), 35, 10 pages (2016).

[5] T. Yoshihisa, T. Hara, "A Low-Load Stream Processing Scheme for IoT Environments", in Proceedings of the 2016 IEEE International Conference on Big Data (Big Data 2016), pp. 263-272 (2016).

[6] N. Chen, Y. Chen, Y. You, H. Ling, P. Liang, and R. Zimmermann,"Dynamic Urban Surveillance Video Stream Processing Using Fog Computing", in Proceedings of the 2016 IEEE Second International Conference on Multimedia Big Data (BigMM 2016), pp. 105-112 (2016).

[7] Q. Ning, C.-A. Chen, R. Stoleru, and C. Chen, "Mobile Storm: Distributed Real-Time Stream Processing for Mobile Clouds", in Proceedings of the 4th IEEE International Conference on Cloud Networking (CloudNet 2015) (2015).

[8] T. Li, J. Tang, and J. Xu, "A Predictive Scheduling Framework for Fast and Distributed Stream Data Processing", in Proceedings of the 2015 IEEE International Conference on Big Data (Big Data 2015), pp. 333-338 (2015).

[9] J.-H. Choi, J. Park, H. D. Park, and O.-G. Min, "DART: Fast and Efficient Distributed Stream Processing Framework for Internet of Things", ETRI Journal, Vol. 39, No. 2, pp. 202-212 (2017).

[10] Y.-K. Kim, Y. Kim, and C.-S. Jeong," RIDE: Real-Time Massive Image Processing Platform on Distributed Environment", EURASIP Journal on Image and Video Processing, 13 pages (2018).

[11] J. Yang, B. Jiang, and H. Song," A Distributed Image-Retrieval Method in Multi-Camera System of Smart City Based on Cloud Computing", Future Generation Computer Systems, Vol. 81, pp. 244-251 (2018).

[12] L. O'Gorman and X. Wang, "Balancing Video Analytics Processing and Bandwidth for Edge-Cloud Networks", in Proceedings of the 24th International Conference on Pattern Recognition (ICPR 2018), pp. 2618-2623 (2018).

[13] K. Kato, A. Takefusa, H. Nakada, and M. Oguchi, "Construction Scheme of a Scalable Distributed Stream Processing Infrastructure Using Ray and Apache Kafka", in Proceedings of the 34th International Conference on Computers and Their Applications (CATA 2019), pp. 368-377 (2019).

[14] Apache Hadoop, available at https://hadoop.apache.org/ (accessed June 1, 2019).

[15] Apache Storm, available at http://storm.apache.org (accessed June 1, 2019).

[16] Apache Flink, available at http://flink.apache.org (accessed June 1, 2019).

[17] Apache Spark Streaming, available at http://spark.apache.org/streaming/ (accessed June 1, 2019).

[18] M. A. Lopez, A. G. P. Lobato, and O. C. M. B. Duarte, "A Performance Comparison of Open-Source Stream Processing Platforms", in Proceedings of 2016 IEEE Global Communications Conference (GLOBECOM 2016), 6 pages (2016).

[19] J. K. Liu, M. H. Au, W. Susilo, K. Liang, R. Lu, and B. Srinivasan,"Secure Sharing and Searching for Real-Time Video Data in Mobile Cloud", IEEE Network, Vol. 29, No. 2, pp. 46-50 (2015).

[20] J. Wang, B. Amos, A. Das, P. Pillai, N. Sadeh, and M. Satyanarayanan, "Enabling Live Video Analytics with a Scalable and Privacy-Aware Framework", ACM Transactions on Multimedia Computing, Communications, and Applications, Vol. 14, No. 3s, Article No. 64 (2018).

(Received November 29, 2019)
(Revised May 7, 2020)

Tomoki Yoshihisa received the Bachelor's, Master's, and Doctor's degrees from Osaka University, Osaka, Japan, in 2002, 2003, 2005, respectively. Since 2005 to 2007, he was a research associate at Kyoto University. In January 2008, he joined the Cybermedia Center, Osaka University as an assistant professor 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-ondemand, broadcasting systems, and webcasts. He is a member of the IPSJ, IEICE, and IEEE.

Satoru Matsumoto received his Diploma's degrees from Kyoto School of Computer Science, Japan, in 1990. He received his Master's degrees from Shinshu University, Japan, in 2004. From 1990 to 2004, he was a teacher in Kyoto School of Computer Science. From 2004 to 2007, he was Assistant Professor of The Kyoto College of Graduate Studies for informatics. From 2007 to 2010, he was Assistant Professor of Office of Society Academia Collabo-ration, Kyoto University. From 2010 to 2013, he was Assistant Professor of Research Institute for Economics & Business Administration, Kobe University. From 2015 to 2016, he was a specially appointed assistant professor of Cybermedia Center, Osaka University. From April 2016 to September 2016, he became a specially appointed researcher. Since November 2016, he became an assistant professor. His research interests include distributed processing systems, rule-based systems, and stream data processing. He is a member of IPSJ, IEICE, and IEEE

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. From April 2015 to February 2020, he was an assistant professor at Nara Institute of Science and Technology. Since March 2020, he has been a senior assistant professor at University of Fukui. His research interests include distributed computing, rule-based systems, and stream data processing. He is a member of the Information Processing Society of Japan (IPSJ) and IEEE.

Yuuichi Teranishi received his M.E. and Ph.D. degrees from Osaka University, Japan, in 1995 and 2004, respectively. From 1995 to 2004, he was engaged Nippon Telegraph and Tele-phone Corporation (NTT). From 2005 to 2007, he was a Lecturer of Cybermedia Center, Osaka University. From 2007 to 2011, He was an associate professor of Graduate School of Information Science and Technology, Osaka University. Since August 2011, He has been a research man-ager and project manager of National Institute of Information and Communications Technology (NICT). He received IPSJ Best Paper Award in 2011. His research interests include technologies for distributed network systems and applications. He is a member of IPSJ, IEICE, and IEEE.